UNIVERSIDAD POLITÉCNICA DE CATALUÑA
Programa de Doctorado:
AUTOMATIZACIÓN AVANZADA Y ROBÓTICA

Tesis Doctoral

Time Series Prediction Using Inductive Reasoning Techniques

Josefina López Herrera

Directores:

François E. Cellier Gabriela Cembrano

Instituto de Organización y Control de Sistemas Industriales

Marzo de 1999

A mis padres por haberme proporcionado la oportunidad de llegar a este nivel acádemico de educación y
a Clemens Porsche Ackermann por el soporte moral y económico que me brindó a lo largo de estos años para realizar esta tesis.

Agradecimientos

Quiero agradecer a los Directores, personal investigador, docente, administrativo, bibliotecario y a mis amigos del Instituto de Informática y Robótica Industrial *(IRI)*, el Laboratorio de Control Automático del Departamento de Automática e Informática Industrial *(ESAII)*, el Instituto de Organización y Control *(IOC)* y la Facultad de Matemáticas y Estadística *(FME)* por haber colaborado siempre para que pudiese desarrollar mi trabajo.

Expreso mi agradecimiento a todas las personas que han contribuido en el trabajo presentado en esta tesis doctoral. Especialmente deseo mencionar a:

Prof. François E. Cellier, Catedrático de la Universidad de Arizona y Director de su programa de Ingeniería de Computación, por la guía y supervisión que me brindó para poder realizar el presente trabajo de investigación. Además de acompañarme siempre en el duro trabajo de investigación me enseñó a disfrutar de los éxitos y aprender de los fracasos, disfruté mucho de su compañía en sus visitas a Barcelona. Le agradezco la oportunidad que me brindó para visitarlo dos veces en la Universidad de Arizona para completar la investigación de esta tesis; así como la hospitalidad y amistad que me brindó junto a Ursula Cellier en mis estancias en Tucson.

Dra. Gabriela Cembrano, Investigadora del Consejo Superior de Investigaciones Científicas *(CSIC)* en el IRI, por haber aceptado a codirigir esta tesis, dar seguimiento y evaluación a los resultados que he ido alcanzando en estos últimos años. Fueron de gran ayuda sus conocimientos de análisis y predicción de series temporales.

Prof. Rafael Huber, Catedrático de la Universidad Politécnica de Cataluña y Director del Instituto de Robótica e Informática Industrial, por haber buscado los recursos para poder desarrollar esta investigación y el interés que siempre ha demostrado por mi trabajo desde que me inició en la investigación de la metodología de Razonamiento Inductivo Borroso a través del curso de doctorado.

Además quiero agradecerle la confianza y el apoyo que me brindó siempre incentivándome a seguir adelante a pesar de las dificultades.

Dra. Pilar Muñoz, de la Facultad de Matemáticas y Estadística *(UPC)*, por haberme proporcionado los datos de la serie temporal de los casos de meningitis, por su colaboración en la investigación de esta tesis y por haberme incentivado junto con el Prof. Manuel Martí Recober a investigar en el complejo mundo de análisis de series temporales a través de los cursos de doctorado.

Dr. Jordi Riera Colomer, Investigador del CSIC en el IRI, por haberme ayudado a solucionar los problemas burocráticos y el apoyo que me proporcionó siendo mi Tutor de estudios.

Dr. Robert Griño, de ESAII e IOC, por su colaboración permanente e incondicional. Por la ayuda prestada en la elaboración del capítulo cuatro de esta tesis proporcionándome la serie de consumo de agua de Barcelona y el mejor modelo neuronal que él publicó con anterioridad para esta serie.

Al personal de los Centros de Cálculo de IRI, IOC y la Universidad de Arizona por haberme proporcionado el soporte técnico que solicité.

Los compañeros de SAPS: Alvaro, Angela, Javier, Jesús, Manel, Paco y Sebastián por los buenos momentos que compartimos en Barcelona.

Los *magníficos* (Albert Castellet, Judith Martínez y Pablo Jiménez), Susana Velázquez y Donald Ballance por haberme ayudado con la configuración del formato de esta tesis.

Gral. de División Marco Tulio Espinosa, Jefe del Estado Mayor del Ejército de Guatemala, por haberme proporcionado siempre su ayuda en cualquier momento y situación.

María del Carmen, Lydia, Marinita y Mike por su amistad y compañía incondicional.

El trabajo de investigación fue parcialmente soportado por:

- TAP96-0882 "Seguridad de funcionamiento en sistemas dinámicos complejos" financiado por la CICYT dentro del Plan Nacional de Tecnologías Avanzadas de la Producción.
- Beca concedida para estancias en el extranjero CIRIT expediente No. 1997BEA12000173.

Resumen

En esta tesis se describen nuevos elementos introducidos en la metodología del *Razonamiento Inductive Borroso (Fuzzy Inductive Reasoning (FIR))* que permiten predecir el comportamiento futuro de series temporales. En la identificación de sistemas ya se habían obtenido antes muy buenos resultados al utilizar esta metodología. Por ello se decidió evaluar esta metodología también en el campo del análisis de series temporales que es un asunto más complejo a causa de la imposibilidad de excitar las entradas de los sistemas que las generan.

Para saber si esta metodología es válida en el campo de análisis de series temporales se hizo un estudio comparativo con otras metodologías como son las conexionistas, las que utilizan modelos lineales y no lineales. Esto permitió caracterizar el tipo de series temporales que mejor predice FIR. Se muestra que esta metodología explota toda la información contenida en los datos disponibles de las series temporales quasi–estacionarias con elementos deterministas.

A causa de la naturaleza cualitativa de la metodología, en un inicio se produjeron predicciones ambiguas. Para superar las dificultades se incorporaron nuevos elementos de predicción. Se modificó la fórmula para calcular la distancia relativa y pesos absolutos de los cinco vecinos más cercanos, se incorporaron nuevas medidas de confianza *similitud y proximidad* que permiten evaluar el error de predicción sin necesidad de conocer el valor real. La medida de *proximidad* se basa en la función de la distancia, mientras que la *similitud* está basada en la similitud de conjuntos borrosos. Se utiliza una generalización de la función clásica de equivalencia basada en las definiciones de cardinalidad y diferencia de la teoría de conjuntos borrosos, originalmente presentada por Dubois y Pradé.

Se desarrollaron dos nuevas técnicas de predicción utilizando las nuevas medidas de confianza que permiten elegir en cada instante de tiempo el mejor modelo cualitativo de predicción.

Estas nuevas técnicas permiten mejorar la predicción de una serie temporal quasi–estacionaria. Al cambiar dinámicamente el modelo cualitativo el error de predicción disminuye considerablemente para una serie temporal no estacionaria con múltiples regímenes.

Se evaluó la relación cuantitativa entre el grado del deterioro de la confianza acumulada y el horizonte de la predictibilidad de una señal demostrándose que la medida cualitativa de *similitud* es más susceptible al error de predicción.

Se presentan también primeros resultados de aplicar esta metodología en el diseño de sensores inteligentes y control predictivo.

Esta tesis se organiza en ocho capítulos y dos apéndices.

En el capítulo uno se describe el enfoque principal de la investigación realizada así como los antecedentes.

En el capítulo dos se establecen los parámetros para clasificar las series temporales que se analizan en esta investigación así como una revisión de todas las metodologías de análisis de series temporales.

En el capítulo tres se presenta la situación actual de la metodología de Razonamiento Inductivo Borroso.

El estudio compartivo de la metodología FIR con las más conocidas en el mundo del análisis de series temporales se describe en el capítulo cuatro.

Se introducen dos nuevas medidas de la calidad de predicción en la metodo-logía FIR. Los resultados de esta investigación se presentan en el capítulo cinco. Se describe la base teórica de estas medidas y se muestran los resultados obtenidos en diferentes tipos de series temporales.

En el capítulo seis se presentan los resultados de aplicar las medidas de calidad de predicción introducidas en el capítulo anterior para mejorar los resultados de predicción en el caso de aplicar FIR en series no estacionarias.

Para evaluar hasta que punto es fiable la predicción, en el capítulo siete se introducen las medidas de calidad de predicción para establecer el horizonte de predicción en series quasi–estacionarias.

En el capítulo ocho se resumen las aportaciones realizadas a la metodologia FIR.

Su aplicación como metodología para diseñar sensores inteligentes y el diseño de controladores predictivos se presentan en los apéndices A y B.

Summary

In this dissertation, new elements are described that have been added to the methodology of *Fuzzy Inductive Reasoning (FIR)*, elements that allow the prediction of the future behavior of time series. In the identification of systems, very good results of using this methodology had been reported earlier. Therefore, it was decided to evaluate the methodology also in the context of predicting time series, a more complex undertaking, because of the impossibility of exerting the systems that generate these time series through their inputs.

In order to determine whether the methodology could be used in the analysis of time series, a comparative study of different methodologies was made, including connectionist methods, as well as linear and non–linear predictors. This study allowed to characterize the types of time series that FIR predicts well. It turns out that FIR exploits all the information that is contained in the available training data of time series that are quasi–stationary with deterministic elements.

Due to the qualitative nature of the methodology, predictions were initially obtained that were ambiguous. In order to overcome these difficulties, new elements of prediction were introduced. The formula used for calculating the relative distances and the absolute weights of the five nearest neighbors was modified, and new confidence measures (based on *similarity* and *proximity*) were incorporated, measures that allow to estimate the prediction error without necessity of knowing the true value of the series. The *proximity measure* is based on a distance function, whereas the *similarity measure* is based on the similarity between fuzzy sets. A generalization of the classical equivalence function is used that is based on definitions of cardinality and difference of the theory of fuzzy sets, originally proposed by Dubois and Pradé.

Two new techniques of prediction were developed that make use of these confidence measures. These methods allow to select, at every time instant, the best qualitative prediction model.

These new techniques allow to improve the prediction of a quasi–stationary time series. By dynamically changing the qualitative model, the prediction error can be reduced considerably in non–stationary time series that operate in multiple regimes.

The relation between the degree of deterioration of the accumulated confidence measure and the horizon of predictability of a signal was evaluated in a quantitative fashion. It was shown that the *similarity measure* is more sensitive to the prediction error than the *proximity measure*.

Also presented are first results obtained when applying the methodology to the problems of the design of intelligent sensors and predictive controllers.

This thesis is structured into eight chapters and two appendices.

In Chapter 1, the principal focus of the investigation is described as well as its antecedents.

In Chapter 2, the parameters are established that allow to classify the time series that are analyzed in this investigation. The chapter also offers a brief review of the methodologies that are being used in time series analysis.

In Chapter 3, the state of the art of the Fuzzy Inductive Reasoning methodology is presented.

A study comparing the performance of FIR with that of the best known time–series prediction methods is presented in Chapter 4.

Two new measures of the prediction quality are introduced in the FIR methodology. The results of this investigation are presented in Chapter 5. The theoretical foundations of these measures are described, and their application to different types of time series is shown.

In Chapter 6, the results of applying the prediction quality measures, introduced in the previous chapter, to the problem of improving the prediction capability of FIR in the case of non–stationary time series are presented.

In order to evaluate up to which point a prediction is reliable, Chapter 7 introduces measures of accumulated prediction quality that can be used to estimate the horizon of predictability in quasi–stationary time series.

In Chapter 8, the contributions obtained in this dissertation related to the FIR methodology are summarized.

Its applications as a methodology for designing intelligent sensors and predictive controllers are presented in Appendices A and B.

Contents

List of Figures

List of Tables

Chapter 1

Introduction, Motivation and Overview.

To be able to predict the future has been a dream of mankind since it became aware of its environment and its ability to manipulate it. People want to "play it safe," by making *informed* decisions, decisions that are based on an understanding of the implications that these decisions will have. This calls for the need to predict the consequences of decisions made, i.e., predict the future.

How can this be accomplished? Essentially, there are two potential roads to success.

1. One can predict the future by *extrapolating* directly from past observations.

2. One can try to make a model that explains relationships between observations made in the past, and then use that model in a simulation to make predictions of the future, given a set of scenarios specified in terms of input trajectories.

Making predictions is *easy*, in fact, it is as easy as throwing a coin. What is difficult, is to know how *good* these predictions are, i.e., estimate the *error* associated with any prediction made.

Direct extrapolation is inherently unsafe, because it does not provide for means that would allow to estimate the quality of the predictions made. Modeling is the better approach, since it allows to correlate different observations with each other, thereby improving the chances of making correct predictions. Also, the modeling approach enables the user to work with input variables, thereby allowing him or her to formulate different

scenarios and observe the consequences that might result when implementing any one of these scenarios.

Yet, also the modeling approach carries inherent risks. Most modeling approaches are *parametric* in nature, i.e., they make use of training data to optimize a set of parameters, then use the model, once trained, for making predictions, without referring back to the training data set.

Any parametric modeling approach presupposes a model structure. Once this decision is made, it will be difficult to assess the errors associated with that decision, i.e., whereas the chosen structure may be appropriate to explain the training data, there is no guarantee that the same structure will also be appropriate to predict outcomes given input data that have never been observed before. The extrapolation capability of the approach lies *precisely* in the structural assumptions made, and is unsafe for the very same reasons.

Luckily, the two approaches outlined above are only two extremes within a continuous spectrum of possible approaches. Is a standard *Box–Jenkins* approach to predicting the future of a univariate time series a modeling or an extrapolation approach? The answer to this question depends probably more on personal tastes than a solid scientific foundation, as the method "models" the time series, but does so directly, i.e., without reasoning about any cause–and–effect relationships, thereby making it impossible to investigate different scenarios[1]. It is a *parametric modeling approach* in the sense that it indeed makes a structural assumption, and then estimates the parameters of that structural model. It is an *extrapolation method* in the sense that it does not identify any system relating inputs to outputs, it only characterizes a signal.

Any successful method will have to somehow exploit the best of both worlds, i.e., use modeling, where applicable, to extract as much information from the available observations as possible, yet use the available training data carefully and cautiously in an extrapolation mode to make sure that the inherent assumptions behind the model do not invalidate the predictions made.

Estimating the error of a prediction is itself a modeling task. A model needs to be made that relates the *testing data* (the new pattern, for which a prediction is to be made) to the *training data* (the patterns observed in the past, the outcomes of which are known). Estimating the error of the prediction essentially means to estimate the relevance that the training data have in explaining the testing data.

Fuzzy Inductive Reasoning (FIR), the methodology investigated in this dissertation, is one such mixed modeling/extrapolation approach. It is

[1] There exist variants of the classical Box–Jenkins method that enable the user to model input/output relationships, but this is not the typical application of the methodology.

a *non–parametric modeling method* that does not presuppose any model structure. All that FIR does in terms of *modeling* is to determine the set of input variables that best explain the observed input/output behavior for the training data. During its *simulation* phase, FIR compares the current testing data (input patterns) with their nearest neighbors in the data base of training data, and interpolates between the previously observed outputs associated with these neighbors.

Investigations into the FIR methodology and its applications are central to the research performed by the *qualitative modeling team* at the *Universitat Politècnica de Catalunya*, a research effort that already led to three Ph.D. dissertations (Nebot 1994; Mugica 1995; de Albornoz 1996). Whereas *Àngela Nebot* (Nebot 1994) dealt with the basic FIR methodology and its application to the qualitative modeling and simulation of ill–defined systems stemming predominantly from the biomedical domain, *Francisco Mugica* (Mugica 1995) treated the systematic design of multivariable fuzzy controllers using FIR, and *Álvaro de Albornoz* (de Albornoz 1996) concentrated on the development of a sister methodology, called *Reconstruction Analysis (RA)*, and applied both methodologies to the problem of fault monitoring in large–scale systems, the present dissertation focuses on *time–series prediction*.

The rationale behind this research is quite simple: in the past, the results published by our research team were often criticized as being *too heuristic*. Although FIR was shown over and over again to produce spectacular results, none of the previous efforts tried to analyze *why* the results were as they were, or *what* are the features that make FIR a successful modeling and simulation technique. Since all three application areas previously dealt with are quite esoteric, it was difficult, if not impossible, to compare the performance of FIR with that of competing technologies.

In contrast, it is *very easy* to come up with methods for predicting time series, and in particular, univariate time series. Yet, making *high–quality* predictions of such time series is a difficult task, exposing weaker methodologies at once as inferior. Thus for the first time, it was attempted to apply FIR to a class of problems where there exist plenty of competitors, such that the quality of FIRs performance can be quantified by comparing it with that of other competing approaches.

It was found that FIR fares well indeed. In all time series tested, FIR performed at least as good as the best among its competitors, and often, FIR outperformed all of the competitors that were used in the comparison.

Yet, comparisons alone do not constitute a Ph.D. dissertation, being as convincing as they may be. The methodological focus of this dissertation is a thorough analysis of techniques for *estimating the error of predictions* made, an aspect of the FIR methodology that had not been looked at before.

This research focus led to a methodology for estimating the *horizon of predictability* of FIR, and a new approach to dynamically choosing between different FIR models based on their own error estimate.

Chapter 2 describes the state of the art of time–series predictions, introducing the ideas behind the various approaches that have been proposed in the past. It classifies time series as well as methods for predicting their future.

Chapter 3 introduces the FIR methodology and discusses the results that had been previously obtained. It then specializes the discussion to the task of predicting either univariate or multi–variate time series. It introduces a new error formula to quantify the errors of predictions made, and it introduces a first set of four time series of different types analyzing how FIR deals with them.

Chapter 4 evaluates the FIR methodology for the purpose of time–series analysis, characterizes different time series, and presents a quantitative comparison of FIR with other contending methodologies.

Chapter 5 discusses means for *estimating the local error* of predictions made, introducing to this end a class of *qualitative confidence measures* that are based in part on estimates of *proximity* and in part on estimates of *similarity* with the nearest neighbors in the input space, it discusses the effects of *dispersion* of the outputs observed for the nearest neighbors in the output space, and it shows the effectiveness of these measures in terms of predicting time series whose outputs are known, so that the predictions using different models can be compared with the true outputs, and the error estimates can be correlated with the true errors.

Chapter 6 describes the dynamic model selection in FIR based on local error estimates. The effectiveness of this technique is demonstrated by means of non–stationary time series, whose behavior changes over time. Different models were constructed for different regimes in which these time series operate, and the dynamic model selection algorithm is used to automatically determine the best model during each time step.

Chapter 7 introduces *global error estimates*, based on the previously introduced local error estimates and assumptions about statistical independence between subsequent local error estimates. Based on these global error estimates, a technique is presented that allows to estimate the *horizon of predictability*, i.e., to determine for how long into the future valid predictions can be made.

Chapter 8 rounds off the dissertation by summarizing the contributions made, and by offering suggestions for future research. Two of these future research directions have already been partially explored by the author of this dissertation. The findings are reported in two appendices that conclude the

dissertation.

Appendix A shows an application of the methodology introduced in Chapter 7, applied to the problem of designing *smart sensors* with look–ahead capabilities. The idea behind the smart–sensor design is straightforward: by the time the sensor detects that the sensed signal passes through a preset threshold and sends out an alarm, it may already be too late to do anything about the problem, forcing the operator to shut down the plant. On the other hand, if the sensors are equipped with local intelligence and can make predictions of the future, such that they can send out an *early warning* when their prediction passes through the threshold, the operator may still have enough time to take corrective action, which would prevent the true signal from ever reaching the threshold.

Appendix B shows an application of the previously introduced smart–sensor technology used in closed–loop operation for the design of a new class of predictive controllers, coined *signal–predictive controllers (SPC)*. The idea behind this research is quite simple: any control action that is based on real measurements comes always late. If the signal that is used to calculate the control action contains a certain degree of lead time, i.e., is based on a prediction, the control may react more quickly, thereby reducing the overshoot. This is similar in nature to adding a D–term to a PI controller, i.e., converting a PI controller to a PID controller, but the results can be slightly better, because the prediction corresponds to a negative delay, rather than a numeric estimate of a derivative, and because the predictor can take non–linearities in the system that generate the signal into account.

Chapter 2

State of the Art of Time–Series Modeling and Simulation

2.1 Introduction

The analysis of time series concerns itself with the investigation of single or multiple observations of measurement data streams taken from a system under observation. It is a characteristic property of time series that they never contain complete information about the system being observed, and in particular, that the excitations that are imposed on the system are not under the observer's control, and are, in many cases, unknown to him or her.

One of the primary objectives of time–series analysis is to be able to predict the future behavior of a measurement signal on the basis of observations of its past behavior.

Time series are assumed to have been generated by dynamic systems. It is therefore important to know when the measurements were taken, i.e., it must either be assumed that the measurement signal has been equidistantly sampled, or alternatively, the time instant when each sample was taken must be stored together with the time series as a second piece of information.

In time–series analysis, it is common to investigate a single data stream stemming from a single source observed by a single sensor. However, it is also common that multiple time series are obtained simultaneously from multiple measurement sensors attached to the same system. In that case, it can be assumed that the individual signals are correlated among each other, and this cross–correlation can be exploited to improve the accuracy of predictions made about the future behavior of these signals.

Time–series analysis has been applied to many different application areas, such as the prediction of financial markets (economical models), or the

monitoring of physiological signals stemming from a patient during surgery (biomedical systems). In engineering, time–series analysis is of interest in the contexts of instrumentation and filtering of signals, and the design of predictive controllers, among others.

There exists a rich literature on methods for analyzing time series. There even were organized competitions on a worldwide level in order to advance the state of the art of methodologies for time–series analysis and prediction (Makridakis and Hibon 1979; Makridakis *et al.* 1984; Weigend and Gershenfeld 1994).

In the early days, most of the models proposed were linear regression models. Their implementation is simple, yet they are quite limited in their capabilities of interpreting time series. They are not capable of dealing with non–linear and/or non–stationary behavioral patterns. They always assume the systems from which the time series has been measured to be linear and to operate under stationary conditions (Priestley 1981; Ljung 1987; Chatfield 1989; Box and Jenkins 1994).

In order to extend the power of time–series analysis to systems with non–linear characteristics, non–linear models were proposed in (Volterra 1959; Tong 1990). In order to be able to deal with time series that exhibit non–stationary characteristics, prefiltering methods were developed that convert non–stationary time series into equivalent stationary ones (Brockwell and David 1991,1996; Box and Jenkins 1994).

The last decade has seen two decisive contributions, made possible as a consequence of the appearance of more powerful computers that allowed to deal with larger data streams and apply more complex algorithms in an interactive fashion. Thanks to these new developments, progress was achieved in statistical modeling techniques (Tong 1990) and in physical modeling methods (Casdagli 1991). These techniques were applied to problems in engineering and control (White and Sofge 1992). They were made possible, because finally, it had become feasible to construct and identify arbitrarily non–linear models rapidly and conveniently.

Some of the more important contributions were the construction and identification of state–space models (Casdagli and Eubank 1992), the application of artificial intelligence to the generation of data–driven rule–based models (Weigend *et al.* 1990), and finally the introduction of learning techniques for model identification (Weigend and Gershenfeld 1994).

The use of *learning techniques* constitutes an important trend in modern time–series analysis methods. It enabled the scientists and engineers to abstract from the explicit equation–driven models of the past to models that are more generic (and usually widely over–parameterized), that make less structural assumptions about the system from which the time series was

generated, and that are therefore more generally applicable to a wider class of time series. The most widely used among these so–called *connectionist models* are *neural networks* that come in many different shades: static (feedforward) networks consisting of static neurons, dynamic (feedback) networks consisting of static neurons, static networks consisting of dynamic (differential equation) neurons, and finally, dynamic networks consisting of dynamic neurons (Kosko 1991,1992).

Another class of connectionist models are those that are based on *fuzzy logic* (Klir and Yuan 1995; Jang 1997). They share many of the properties of neural network models, yet their internal structure is quite different. Some of these techniques are *non–parametric*, i.e., they refer to the training data themselves during the prediction process, rather than incorporating the knowledge contained in the training data in a set of model parameters. Some of these techniques use *model synthesis* methods rather than *model training* approaches. The methodology advocated in this dissertation, *Fuzzy Inductive Reasoning (FIR)* falls into this category of approaches. FIR models are qualitative non–parametric methods that are synthesized rather than trained.

There also exist a number of mixed methods involving either fuzzy inferencing systems with parameters determining the shape of the fuzzy membership functions that are trained using neural networks, or neural networks, the weights of which are identified using fuzzy logic (Takagi and Sugeno 1991; Stahl 1996; Ghoshray 1996; Zhang and Li 1996; Chen 1996; Ishikawa and Moriyama 1996; Burr 1998).

Finally, some references distinguish between the *prediction* of time series and their *simulation* (Ljung 1987; MathWorks 1997). Prediction methods are simple data extrapolation techniques that do not rely on generating a model first, but simply use the available data to make predictions about the future. These are single–shot approaches. They do not make use of previously made predictions in further predictions. On the other hand, simulation methods are based on (either explicit or implicit) models of the time series. They first create a model, then make predictions using that model. They are often recursive in nature, i.e., they make use of previous predictions in making more predictions further into the future.

This dissertation deals with *qualitative simulation of time series*. The proposed methods are all based on *qualitative models* synthesized by use of a training data set.

Sometimes, time series analysis is used for purposes other than determining future behavior. For example, different signatures of a foreign submarine in muddy waters constitute multiple correlated time series that are being analyzed with the purpose of identifying where the submarine currently

is. This dissertation does not deal with these types of time series analyses.

2.2 Characterization of Time Series

Weigend and Gershenfeld (1994) provided a useful classification of different time series. In Table 2.1, it is repeated with a few modifications and enhancements.

Table 2.1: Classification of Time Series

natural		synthetic
stationary		non–stationary
time invariant		time varying
low dimensional		stochastic
clean		noisy
short		long
dormant		active
documented		blind
linear		non–linear
scalar		vector
single recording		multiple recordings
continuous		discrete

In academia, time series are quite often synthesized from simulation experiments. Such time series are very clean, and techniques that are capable of dealing with such synthetic time series are not necessarily also well suited to deal with natural (measured) time series. Hence it is important to record whether a time series is *natural* or *synthetic*. This dissertation shall deal mostly with natural time series, as the methodology embraced in this dissertation is well suited to filter out residual noise components (Ljung 1992; Ivanova *et al.* 1994; Weigend and Mangeas 1995).

Many, especially statistical, techniques rely on *stationary* behavior. If a time series is *non–stationary*, they have difficulties dealing with it. Also, many non–parametrical techniques (such as FIR) have difficulties dealing with growth functions, as they can only predict behavioral patterns that have been previously observed as part of the training series. Although a work–around has been found (Moorthy *et al.* 1998), this dissertation shall deal with stationary or pseudo–stationary time series only.

Time series can be either *time invariant* or *time varying*. A time varying time series is one that operates in different regimes at different points in

time (Chang 1996). For example, the signature of a military aircraft flying at high–altitude horizontal flight is quite different from that of the same aircraft during attack. In this dissertation, a methodology will be introduced that is particularly geared to dealing successfully with time series that are time varying, i.e., that operate in different regimes during different time periods.

Time series can be *low dimensional*, exhibiting a limited set of behavioral patterns, such as periodicity, or *high dimensional*, leading to chaotic behavior. It shall be shown in this dissertation that FIR is well suited to deal with both "deterministic" and "stochastic" time series[1].

A *noisy* time series is frequently produced by problems with the measurement equipment. For example, the ECG signal of a patient may be interrupted, because the nurse takes off the sensor while she cleans the patient, or may be disturbed, because the patient moves around in his or her bed. Similarly, industrial processes, such as the water supply system of a region, may be equipped at times with faulty sensors that cannot be immediately replaced due to the geographically distributed nature of the system. Most natural time series are somewhat noisy. Statistical and fuzzy techniques are particularly well suited to filter out measurement noise (Klir and Folger 1988; Lee 1990; Karr and Gentry 1993; Tanaka *et al.* 1995; Wang and Langari 1995; Kim and Kim 1997).

Time series can be *short* or *long*. A short time series may be caused by a transitory event, such as a surgery, that is of limited duration. Although it is possible to increase the sampling rate, thereby making the time series "longer," this may not help. There exists a natural sampling rate for each time series that is related to the natural frequencies (eigenfrequencies) of the system from which the time series is sampled. Oversampling increases the length of the time series, but not its information contents. Forecasting techniques that are based on models suffer from *data deprivation* in the presence of short or oversampled time series. The simple extrapolation techniques may work best in this case, at least for single time series. Multiple correlated time series may still be better predicted using model–based approaches.

A time series can be *active* or *dormant*. These terms relate to the type of excitation placed on the inputs of the system from which the time series is drawn. Active time series are more easily identifiable, since they show all patterns that the system is capable of exhibiting. Yet, dormant time series

[1]The term "stochastic time series" is somewhat a misnomer as deterministic systems (which most real systems are) with deterministic inputs (a meaningful assumption) can never produce stochastic outputs. Thus, "high–dimensional systems" or "chaotic patterns" are better terms in a puristic sense. However, the term "stochastic system" is commonly used, and therefore, it shall not be strictly avoided in the context of this dissertation either.

are often desirable and unavoidable. For example, a surgeon is very happy if his or her patient exhibits "dormant" behavior throughout the surgery. It is certainly unacceptable to "exert" a patient unnecessarily for the purpose of obtaining a time series that is more easily identifiable. No special efforts were made in this dissertation to avoid dealing with time series that are poorly excited, yet most of the time series used in this dissertation are naturally well excited.

A time series may be *documented* or *blind*. These terms relate to the amount of knowledge available about the systems from which the time series was drawn. Clearly, such knowledge can be exploited, when available, and especially *deductive* approaches (which are of no immediate concern to this dissertation) make use of such knowledge to infer a model structure that matches that of the underlying system. Since FIR is a strictly *inductive* method, the distinction is of little concern to this dissertation.

The system from which the time series is drawn may be either *linear* or *non–linear*. Linearity is being exploited mostly by some of the classical techniques, and is therefore of little concern to this dissertation.

Time series can either be of the *scalar* or of the *vector* type. These terms relate to the number of correlated time series observed from the system under investigation. A scalar time series is one that consists of a single trajectory, whereas a vector time series consists of multiple correlated trajectories. FIR can easily deal with both situations, although this dissertation concerns itself mostly with scalar time series, as they are more difficult to predict.

Time series may consist of a *single recording*, or of *multiple recordings*. Multiple recordings lead to multiple uncorrelated trajectories representing different patterns of the same phenomenon. FIR is well suited to deal with both single and multiple recordings, and examples of both types shall be discussed in this dissertation.

Finally, the system from which the time series is drawn can be either *continuous* or *discrete*. Since the time series itself is always sampled, the distinction is of not much concern to the discussion at hand.

2.3 Classification of Time Series Analysis Techniques

A first coarse classification can be made by distinguishing between *prediction* and *simulation* approaches, i.e., techniques that operate on the time series directly *vs.* techniques that first create a model and then operate on that model. Yet, this classification is not truly crisp. Even simple extrapolation

techniques usually identify parameters of a polynomial or regression function. Whether this polynomial or regression function is called a "model" is simply a question of taste.

A second (still very coarse) classification can be made by distinguishing between *deductive* and *inductive* modeling techniques. Again, also this classification is not strictly crisp. In fact, there are no strictly deductive approaches to time–series analysis. Even in the case of an extremely well documented time series, such as the Lorenz attractor series described in (Gershenfeld and Weigend 1994), a "deductive" modeling approach would make use of the knowledge provided about the system to conclude that the output signal is governed by the Lorenz equations, a set of three very simply and well understood bi-linear differential equations that are furthermore autonomous, i.e., excitation free. The modeler would thus only need to identify the three (linear) parameters of the Lorenz model, such that they match optimally well the observed output patterns. The technique is almost purely deductive, except for the identification of the parameters, which can be considered an inductive process. The less structural assumptions are being made about the system from which the time series is drawn, the more the approach must be considered inductive. FIR is an essentially purely inductive modeling approach.

Among the various available primarily inductive modeling approaches, the following four: *Fuzzy Inductive Reasoning (FIR)*, (López *et al.* 1996), *Artificial Neural Networks (ANN)* (Kosko 1991; Kosko 1992; Wan 1994), *Box–Jenkins (BJ)*, (Box and Jenkins 1994), and *NARMA (NRM)*, (Connor *et al.* 1992) shall be used in the subsequent chapters of this dissertation, although only FIR shall be discussed in any great detail. For this reason, it makes sense to analyze how the four approaches can deal with the different types of time series presented in the previous section. Table 2.2 presents an overview of these classification.

ANNs and NARMA (NRM) models can better deal with non–stationary behavior, because they do not exploit stationarity explicitly, except that training the weights of the neurons may become more problematic in the case of a non–stationary time series. FIR can deal with non–stationary behavior, such as growth functions, but the data need to be prefiltered (Moorthy *et al.* 1998). Box–Jenkins (ARIMA, BJ) models are statistical models that are based on an assumption of strict stationarity.

A technique shall be introduced in Chapter 6 of this dissertation that allows FIR to deal elegantly and conveniently with systems that operate in different regimes. All other techniques have difficulties dealing with time varying systems, although applications of neural networks that switch between different modes have been described in the literature (Weigend and

Nix 1994; Weigend *et al.* 1995; Guo *et al.* 1997; Kim and Kim 1997; Papadakis *et al.* 1998).

The Box–Jenkins models have difficulties when dealing with periodic behavior. Special techniques must be applied when dealing with periodic behavioral characteristics. The reasons for this statement will become clear in Chapter 4 of this dissertation, where the Box–Jenkins approach to time–series analysis is explained in more detail. None of the other techniques have difficulties of this kind.

High dimensionality leads to behavior that can be interpreted as stochastic. ANN and NARMA models have more difficulties when dealing with stochastic behavior. They do filter out noise, but FIR and the Box–Jenkins models actually exploit the statistical properties of the noise signals, whereas ANN and NARMA only try to get rid of the noise.

FIR has more difficulties than any of the other techniques when faced with data deprivation. This is due to the inherent complexity of the approach. The simpler a method is, the less data it needs to produce results (although the results generated may often not be reasonable as a consequence of the data deprivation problem).

Neither FIR nor ANN techniques explicitly exploit the structural knowledge available. They are purely inductive, whereas the other two approaches are partly deductive. Box–Jenkins makes an assumption of strict linearity, and NARMA has a predefined (rather simple) structure that allows it to also exploit structural knowledge to some extent. The very same characteristic makes it impossible for Box–Jenkins methods to deal with non–linear systems in any decent way (except by ignoring the non–linearity), and makes it harder for NARMA to deal with arbitrarily non–linear systems.

Both the Box–Jenkins approaches and the NARMA methods have more difficulties when dealing with multi–variable systems, because it increases the complexity of their models. ANN methods do not have this limitation, and FIR models even work better when applied to multi–variable systems.

One of the most important advantages of FIR is its ability to generate decent models of time series in a fairly automated manner. The parameters that need to be chosen are either quite intuitive (such as the mask depth (cf. Chapter 3)) or fairly insensitive (such as the number of classes (cf. Chapter 3)), and their selection could therefore be fully automated. In contrast, Box–Jenkins and NARMA models require a lot of user intervention, and the decisions to be taken are often non–trivial, i.e., require much knowledge about the specific characteristics of the time series to be predicted and/or the modeling methodology in use. ANN models are less sensitive to user intervention, but require parameters to be user selected that are non–intuitive and sensitive, such as the number of network layers and the number

of neurons per layer, which then often leads to quite a bit of trial and error, before the ANN offers its best performance.

2.4 Conclusions

This chapter provided an introduction to the nature and importance of time series analysis. It also offered an overview over and a classification of the different types of time series to be found. It finally suggested a classification of some of the more commonly used inductive modeling approaches for time series.

There are other aspects of these techniques that were not mentioned in this brief introduction. For example, FIR has a technique built into the methodology that enables it to generate not only an estimate of future values of the time series, but with it an estimate of the *error* associated with the aforementioned estimate. This is easily the single most important characteristic of FIR, a facet of the methodology that will be extensively discussed and exploited in subsequent chapters of this dissertation.

Table 2.2: Classification of Inductive Time Series Analysis Methods

Type of Time Series		Type of Modeling Method			
		FIR	ANN	BJ	NRM
Behavior	*stationary*	**	**	**	**
	non-stationary	*	**	-	**
Regimes	*single*	**	**	**	**
	multiple	*	*	-	-
Dimensionality	*low dimensional*	**	**	*	**
	high dimensional	**	*	**	*
Source	*clean*	**	**	**	**
	noisy	**	*	**	*
Length	*short*	*	**	**	**
	long	**	**	**	**
Excitation	*dormant*	*	*	*	*
	active	**	**	**	**
Knowledge	*exploits structure*	-	-	**	**
	blind	**	**	-	*
Linearity	*exploits linearity*	-	-	**	**
	non-linear	**	**	-	*
Sensors	*single*	**	**	**	**
	multiple	**	**	*	*
Observations	*single*	**	**	**	**
	multiple	**	**	**	**
System type	*continuous*	**	**	**	**
	discrete	**	**	**	**
Modeling	*designer intervention*	-	*	**	**
	automatic	**	*	-	-
Legend	*well suited*		**		
	suited		*		
	unsuited		-		

Chapter 3

Fuzzy Inductive Reasoning for Time Series Prediction

3.1 Introduction

In this chapter, the application of *Fuzzy Inductive Reasoning (FIR)* to time–series analysis and forecasting is presented. The methodology combines facets of *Inductive Reasoning* and *Fuzzy Logic*.

The FIR methodology operates strictly on measured data streams, and reasons about the spatial and temporal relationships among these data without pre–proposing any equation structure. In this thesis, FIR is being used to predict the future behavior of either univariate or multivariate time series, i.e., it learns and analyzes observed patterns of measurement signals, and predicts their future behavior on the basis of their own past, without ever identifying the systems from which these signals were generated.

The Inductive Reasoning technique was originally suggested by G. Klir of the State University of New York at Binghamton (Uyttenhove 1978; Klir 1985). It was reimplemented in CTRL–C by F. Cellier at the University of Arizona (Cellier and Yandell 1987; Cellier 1987). Fuzzy measures were added to the originally crisp inductive modeling methodology by D. Li (Li and Cellier 1990). Several additional features were added to the FIR methodology in due course, such as the treatment of missing values (Nebot 1994) and measures for estimating the prediction error (Cellier *et al.* 1998). FIR has meanwhile been ported from CTRL–C to Matlab (Cellier *et al.* 1996). This is the form in which the software is currently being used.

Many interesting and promising results in dynamic system identification using FIR were reported in recent years. Among others, research related to this methodology already led to three Ph.D. dissertations describing

methodological contributions to and applications of the FIR methodology related to dynamical system identification. Nebot (1994) discusses the use of FIR for identifying ill–defined systems, primarily in the context of biomedical applications; Mugica (1995) describes the use of FIR for the systematic design of fuzzy controllers; and de Albornoz (1996) analyzes the use of FIR for detecting and characterizing faults in large–scale systems.

Time–series analysis is different from the mathematical modeling and simulation of dynamical systems in several respects. Although a time series can be interpreted as an output of a system, it is, by definition, an output of an *unknown* system. Neither the system characteristics nor the input functions driving the system are known, and consequently, time–series analysis must content itself with estimating future output values by means of extrapolation from their own past. Although the prediction of (at least univariate) time series is conceptually simpler than the identification and simulation of dynamical systems, with the result that many methodologies have been proposed that can be used to predict univariate time series, but that cannot be used to identify multi–variate time series and/or dynamical systems, the prediction of univariate time series poses serious practical problems because of the lack of information available, and it can be quite a bit more difficult to obtain good results for time–series prediction than for dynamical system identification and simulation.

The scientific community has been interested in the analysis of time series for a long time. Time series are important when studying the behavior of a system that is not completely understood, such as the time-varying intensity of a star, or the stock market. Clearly, one should not expect similarly accurate results when forecasting a time series as when simulating a known system with known inputs. It is also important to recognize that the time horizon of a meaningful prediction will, in this case, usually be limited, and in fact, may be rather short. Furthermore, a successful prediction of a time series depends on the characteristics of the time series. It is to be expected that a stationary or quasi–stationary process can be predicted better and over a longer time horizon than a non–stationary process. Also, time series that exhibit a more regular, more deterministic (e.g., cyclic) behavior should be more easily predictable than time series that exhibit a more stochastic behavior.

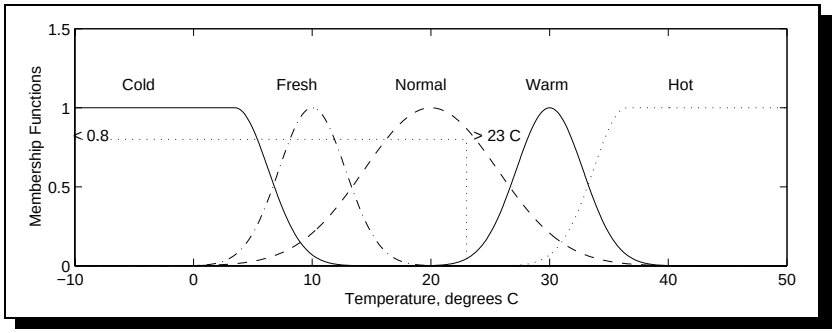

Figure 3.1: Fuzzification process

3.2 The Fuzzy Inductive Reasoning Methodology

Fuzzy inductive reasoning *FIR* is a modeling and simulation methodology that generates a qualitative input/output model of a system by finding the best possible fuzzy finite state machine between discretized (fuzzified) input and output states of the system. The methodology is composed of four main functions that will be described in the remainder of this section.

3.2.1 Fuzzification

The process of converting quantitative (real–valued) variables into qualitative triples is normally referred to as *recoding* in the FIR methodology. The first component of the triple is the *class value*, the second is the *fuzzy membership function value*, and the third is the *side value* (Cellier 1991). The process of fuzzy recoding (or fuzzification) is illustrated in Figure 3.1.

This example fuzzifies ambient temperature using five classes. A quantitative value of 23^o C is recoded into the class value "normal" with a fuzzy membership value of 0.8 and a side value of "right." The side value refers to the fact that the quantitative value is situated to the right or left of the peak of the Gaussian membership function associated with the selected class. Clearly, no information is being lost in the process. The original quantitative value can be regenerated (defuzzified) easily and unambiguously from the qualitative triple. In the current implementation of FIR, a Matlab (MathWorks 1997) toolbox, classes are denoted by positive integers rather than linguistic variables, and the side value is an integer in the range $\{-1, 0, +1\}$, where -1 stands for *left*, $+1$ represents *right*, and 0 denotes *center*.

The name "recoding" had been chosen before fuzzy measures were added to the FIR methodology, i.e., when this was purely a process of *discretizing* real–valued (continuous) variables into class–valued (discretized) variables. Today, the name "fuzzification" would seem more adequate. Yet, the former name is still appropriate, because FIR, contrary to most other fuzzy techniques, does not make use of the information stored in the tails of the fuzzy membership functions. Each quantitative value corresponds to exactly one qualitative triple, rather than several qualitative pairs of class and membership values. The added *side value*, which is peculiar to the FIR methodology, makes this possible. Hence fuzzification (or recoding), in the context of FIR, entails nothing but a non–linear mapping of one space into another equivalent space.

The process of recoding is applied to each observed variable (trajectory) separately. The recoded qualitative *episodical behavior* is stored in three matrices, one containing the class values, the second storing the membership function values, and the third keeping the side values. Each column of these matrices represents one of the observed variables, and each row represents one recorded state. The trajectory behavior can thus be separated into a set of trajectories, y_i, as shown in the following example:

$$\begin{array}{ll} time & \begin{array}{cc} y_1 & y_2 \end{array} \\ \begin{array}{l} 0.0 \\ \delta t \\ 2\delta t \\ 3\delta t \\ \vdots \\ (n_{\mathrm{rec}}-1)\cdot\delta t \end{array} & \left(\begin{array}{cc} \ldots & \ldots \\ \ldots & \ldots \\ \ldots & \ldots \\ \ldots & \ldots \\ \vdots & \vdots \\ \ldots & \ldots \end{array}\right) \end{array} \tag{3.1}$$

In the above example, it was assumed that two separate signals are measured simultaneously from the same system, resulting in two (most likely correlated) trajectories that can be interpreted as a multivariate time series. The sampling rate, δt, needs to be chosen in accordance with the eigenfrequencies of the signals to be observed (Shannon sampling theorem).

For many practical applications, it has been found that the optimal number of classes is between three and five (Cellier 1991). More classes provide a finer resolution, but also call for more training data in order to provide relevant history information.

The fuzzy membership value will be calculated, in accordance with the most relevant among the Gaussian membership functions (Figure 3.1), as follows:

$$Memb_i = \exp(-k_i \cdot (x - \mu_i)^2) \tag{3.2}$$

where x it is the variable to be fuzzified, i is the index of the most relevant Gaussian, i.e., the index representing the selected class, μ_i is the numerical value corresponding to the center of the chosen class, which is also the algebraic mean of the two *landmarks* that separate the chosen class from its left and right neighbors, and $-k_i$ shapes the fuzzy membership function of the chosen class such that it decays to a value of 0.5 at the two landmarks that mark the left and right limits of the interval associated with the chosen class.

From statistical considerations, it is known that in any class analysis, one would like to record each possible discrete state at least five times (Law and Kelton 1990). Thus, a relation exists between the possible number of legal states and the number of data points that are required to base the modeling effort upon:

$$n_{\text{rec}} \geq 5 \cdot n_{\text{leg}} = 5 \cdot \prod_{\forall i} k_i \tag{3.3}$$

where n_{rec} denotes the total number of recordings, i.e., the total number of observed states, n_{leg} denotes the total number of different legal (discrete) states, i is an index that loops over all variables belonging to the observation, and k_i denotes the number of classes associated with the i^{th} variable. The number of variables is usually given, and the number of recordings is frequently predetermined. In such a case, the optimum number of levels can be determined from the following equation:

$$n_{\text{lev}} = \text{round}\left(\sqrt[n_{\text{var}}]{\frac{n_{\text{rec}}}{5}} \right) \tag{3.4}$$

For reasons of symmetry, an odd number of levels is often preferred over an even number of levels. The number of levels of the variables determines the expressiveness and predictiveness of the qualitative model. The *expressiveness* of a qualitative model is a measure of the information content that the model provides. The *predictiveness* of a qualitative model is a measure of its forecasting power, i.e., it determines the length of time over which the model can be used to forecast the future behavior of the underlying system (Li and Cellier 1990).

3.2.2 Qualitative Modeling

Once the quantitative trajectory behavior has been recoded into a qualitative episodical behavior, the process of modeling consists of finding finite

automata relations between the recoded variables that make the resulting state transition matrices as deterministic as possible. Such a relation is called a *mask*. An example of a mask related to the previously introduced multivariate time series might be:

$$\begin{array}{c|cc} {}_t\backslash^x & y_1 & y_2 \\ t-5\delta t & -1 & 0 \\ t-4\delta t & 0 & -2 \\ t-3\delta t & 0 & 0 \\ t-2\delta t & -3 & 0 \\ t-\delta t & -4 & -5 \\ t & 0 & +1 \end{array} \tag{3.5}$$

The negative elements in this matrix denote inputs of the qualitative functional relationship, so-called m–inputs. The above example has five m–inputs. The positive value represents the m–output. A mask denotes a dynamic relationship between qualitative variables. A mask has the same number of columns as the episodical behavior to which it is applied, and it has a certain number of rows. The number of rows of the mask matrix is called the *depth* of the mask. The above mask would denote the structural relationship:

$$y_2(t) = \mathrm{f}\left(y_1(t-5\delta t), y_2(t-4\delta t), y_1(t-2\delta t), y_1(t-\delta t), y_2(t-\delta t)\right) \tag{3.6}$$

A *mask candidate matrix* is an ensemble of all possible masks, from which the best one is chosen by a mechanism of exhaustive search. The mask candidate matrix contains -1 elements where the mask has a potential m–input, it contains a $+1$ element where the mask has its m–output, and it contains 0 elements to denote forbidden connections. Thus, a mask candidate matrix for the previous two–variable example might be:

$$\begin{array}{c|cc} {}_t\backslash^x & y_1 & y_2 \\ t-5\delta t & -1 & -1 \\ t-4\delta t & -1 & -1 \\ t-3\delta t & 0 & 0 \\ t-2\delta t & -1 & -1 \\ t-\delta t & -1 & -1 \\ t & 0 & +1 \end{array} \tag{3.7}$$

Each of the possible masks is compared to the others with respect to its potential merit. The optimality of the mask is evaluated with respect to the maximization of its forecasting power. The Shannon entropy measure is used

to determine the uncertainly associated with the forecasting of the output state, given a feasible input state.

The Shannon entropy relative to one input is calculated from the equation

$$H_i = \sum_{\forall o} \mathrm{p(o|i)} \cdot \log_2 \mathrm{p(o|i)}, \tag{3.8}$$

where $p(o/i)$ is the likelihood of a certain output state o to occur, given that the input state i has already occurred. The likelihood is to be understood, in the usual sense of fuzzy logic, as the confidence expressed in the occurrence of the outcome o relative to the occurrence of any other outcome. It is computed from the fuzzy membership information associated with the observations contained in the experience data base.

The overall entropy of the mask is then calculated as the sum

$$H_m = -\sum_{\forall i} \mathrm{p_i} \cdot \mathrm{H_i}, \tag{3.9}$$

where p_i is the likelihood of that input to occur. The highest possible entropy $H_{\max}$ is obtained when all likelihoods are equal, and a zero entropy is encountered for relationships that are totally deterministic.

A normalized overall entropy reduction H_r is then defined as:

$$H_r = 1.0 - \frac{\mathrm{H_m}}{\mathrm{H_{max}}} \tag{3.10}$$

H_r is obviously a real number in the range between 0.0 and 1.0, where higher values usually indicate an improved forecasting power. A performance indicator that has this property is called a *quality measure* (Cellier 1991). Quality measures are useful in multi–criteria optimizations, since a meaningful overall performance index for such an optimization can be defined as the product of the quality measures representing the (often competing) individual criteria.

The optimal mask among a set of mask candidates could be defined as the one with the highest entropy reduction. However, a problem remains. If the complexity of the mask is increased, the state transition matrix becomes more and more deterministic. With growing mask complexity, more and more possible input states (combinations of levels of the various input variables) exist. Since the total number of observations n_{rec} remains constant, the observation frequencies of the observed states will become smaller and smaller. Very soon, a situation will be found where every state that has ever been observed has been observed precisely once. This leads obviously to a completely deterministic state transition matrix. Yet the predictiveness of the model may still be very poor, since already the next predicted state has

probably never before been observed, and that means the end of the forecast. Therefore, this consideration must be included in the quality measure.

It was mentioned earlier that, from a statistical point of view, one would like to make sure that every state is observed at least five times. This demand leads to the definition of an *observation ratio* (Li and Cellier 1990):

$$OR = \frac{5 \cdot n_{5\times} + 4 \cdot n_{4\times} + 3 \cdot n_{3\times} + 2 \cdot n_{2\times} + n_{1\times}}{5 \cdot n_{\text{leg}}} \tag{3.11}$$

where:

n_{leg} = number of legal input states;
$n_{1\times}$ = number of input states observed only once;
$n_{2\times}$ = number of input states observed twice;
$n_{3\times}$ = number of input states observed thrice;
$n_{4\times}$ = number of input states observed four times;
$n_{5\times}$ = number of input states observed five times or more.

If every legal input state has been observed at least five times, OR is equal to 1.0. If no input state has been observed at all (no data), OR is equal to 0.0. Thus, OR also qualifies as a quality measure.

The *mask quality* is consequently defined as the product of the uncertainty reduction measure and the observation ratio:

$$Q = \mathrm{H_r} \cdot \mathrm{OR} \tag{3.12}$$

The *optimal mask* is the mask with the largest Q value.

3.2.3 Qualitative Simulation

Once the optimal mask has been determined, it can be applied to the given fuzzified time series resulting in an *input/output history* consisting of three matrices, one each for the class values, membership values, and side values of the input/output observations. This process is depicted in Figure 3.2.

On the left side, an excerpt of the recoded class–value matrix is shown with Mask (3.5) laid over the first five rows. The square boxes denote the positions of the m–inputs, whereas the angular brackets denote the position of the m–output. The m–inputs and m–output are read out from the class–value matrix from left to right and top to bottom. They are written next to each other into a row of the input/output matrix that is shown on the right side of Figure 3.2. The variables i_j here denote the five m–inputs, whereas the variable o_1 denotes the single m–output. The mask is then shifted down by one row, and the procedure is repeated, leading to the second row of the input/output matrix, etc. In this way, the dynamic relationship between the

$$
\begin{array}{c} \\ 0 \\ \delta t \\ 2\delta t \\ 3\delta t \\ \Downarrow\ 4\delta t \\ 5\delta t \\ 6\delta t \\ 7\delta t \end{array}
\begin{array}{c} y_1 \\ \left(\begin{array}{c} \boxed{1} \\ 2 \\ 1 \\ \boxed{2} \\ \boxed{3} \\ 2 \\ 3 \\ 1 \end{array}\right. \end{array}
\begin{array}{c} y_2 \\ \begin{array}{c} 2 \\ \boxed{3} \\ 2 \\ 2 \\ \boxed{3} \\ <1> \\ 1 \\ 3 \end{array}\left.\begin{array}{c}\\\\\\\\\\\\\\\end{array}\right) \end{array}
\Longrightarrow
\begin{array}{c} \\ 5\delta t \\ 6\delta t \\ 7\delta t \end{array}
\begin{array}{c} i_1 \quad i_2 \quad i_3 \quad i_4 \quad i_5 \quad o_1 \\ \left(\begin{array}{cccccc} \boxed{1} & \boxed{3} & \boxed{2} & \boxed{3} & \boxed{3} & <1> \\ 2 & 2 & 3 & 2 & 1 & 1 \\ 1 & 2 & 2 & 3 & 1 & 3 \end{array}\right) \end{array}
$$

Figure 3.2: Flattening dynamic relationships through masking

values contained in the original recoded class–value matrix can be flattened out.

Since the input/output matrix contains functional relationships within single rows, the rows of the class–value matrix belonging to the input/output history can now be sorted in alphanumerical order, while treating the other two matrices as tags. The result of this operation is called the *input/output behavior* of the system. The input/output behavior also consists of three matrices, together defining a fuzzy finite state machine. For each combination of input values, it shows, which outputs are likely to be observed.

The input/output behavior can now be used in predictions. Given a new input state (a combination of values of all m–inputs), the input/output behavior (sometimes also referred to as the *experience data base*) can be searched for similar m–input patterns having been observed in the past. To this end, a *position vector* is associated with each input state in the input/output behavior. Each qualitative triple is thereby associated with a quantitative value that can be viewed as a *normalized defuzzification* of that triple.

In the normalization, each Gaussian is mapped separately to the range $[-0.5, +0.5]$, thus $\mu_i = 0.0$. k_i can be determined by evaluating Equation (3.2) at one of the borders, e.g. by setting $x = 0.5$ and $Memb_i = 0.5$:

$$0.5 = \exp\left(-k_i \cdot (0.5^2)\right) \tag{3.13}$$

i.e.,

$$k_i = -4.0 \cdot \ln(0.5) \tag{3.14}$$

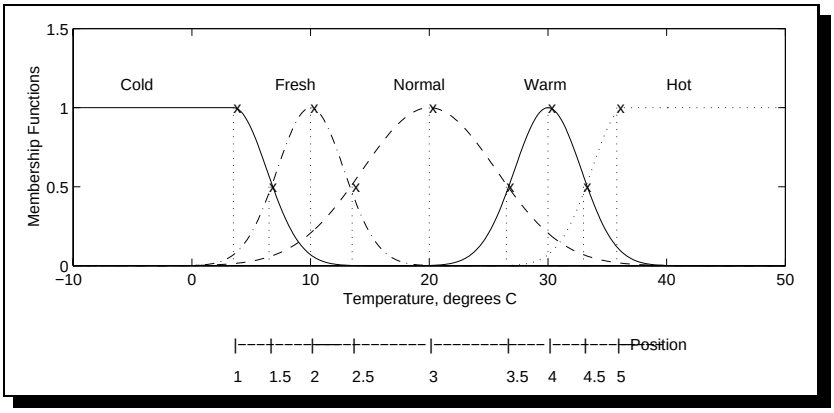

Figure 3.3: Position value of the normalized defuzzification.

thus:

$$Memb_i = \exp\left(4.0 \cdot \ln(0.5) \cdot (x^2)\right) \tag{3.15}$$

or:

$$x = side_i \cdot \sqrt{\frac{\ln(Memb_i)}{4.0 \cdot \ln(0.5)}} \tag{3.16}$$

Using this normalized defuzzification, the position value, pos_i associated with a given qualitative triple, $\{class_i, Memb_i, side_i\}$, can be defined as:

$$pos_i = class_i + side_i \cdot B \cdot \sqrt{-\ln(Memb_i)} \tag{3.17}$$

where:

$$B = \sqrt{\frac{-1.0}{4.0 \cdot \ln(0.5)}} \tag{3.18}$$

The process of determining the position value is depicted in Figure 3.3.

It can be seen that the relationship between the original quantitative value, x, and the normalized position value, pos_i, is a (usually non–linear) transformation that maps the definition range of x into the range $[1.0, n_{\text{cl}}]$, where n_{cl} is the number of classes associated with the fuzzification of x.

The position vector associated with an input state is the vector of the position values of the individual variables associated with the input state:

$$\mathbf{pos_{in}} = [pos_1, pos_2, \ldots, pos_n] \tag{3.19}$$

assuming that the input state contains n variables, i.e., that the optimal mask that was used to produce the input/output behavior contains n m–inputs, i.e., is of complexity $(n+1)$.

A position vector is associated with the current input state for which an output is to be predicted, and also with every input state in the input/output behavior.

Let $\mathbf{pos_{in}}$ denote the position vector of the current input state, and $\mathbf{pos_{in}^{j}}$ the position vector of the j^{th} entry in the input/output behavior. The distance between the current input state and any input state in the experience data base can be defined as:

$$dis_{\text{in}}^{j} = \|\mathbf{pos_{in}} - \mathbf{pos_{in}^{j}}\| \tag{3.20}$$

It is now easy to determine the closest neighbor in the experience data base. The class and side values of the current output are simply predicted to be the same as those of the nearest neighbor.

Prediction of the membership value of the current output proceeds differently. To this end, the five nearest neighbors are determined. The membership value of the current output is predicted as a weighted sum of the membership values of the outputs of the five nearest neighbors in the experience data base.

In order to prevent a possible division by zero in the proposed algorithm, it is necessary to avoid distance values of 0.0:

$$d^{j} = \max(dis_{\text{in}}^{j}, \epsilon) \tag{3.21}$$

where ϵ is the smallest number that can be distinguished from 1.0 in addition.

$$s_d = \sum_{j=1}^{5} d^{j} \tag{3.22}$$

is the sum of the distances of the five nearest neighbors, and:

$$d_{\text{rel}}^{j} = \frac{d^{j}}{s_d} \tag{3.23}$$

are the relative distances.

Absolute weights are computed as:

$$w_{\text{abs}}^{j} = \frac{1.0}{d_{\text{rel}}^{j}} \tag{3.24}$$

and:

$$s_w = \sum_{j=1}^{5} w^j_{\text{abs}} \tag{3.25}$$

is the sum of the absolute weights. Hence relative weights can be computed as:

$$w^j_{\text{rel}} = \frac{w^j_{\text{abs}}}{s_w} \tag{3.26}$$

and the membership value of the current output is:

$$Memb_{\text{out}} = \sum_{j=1}^{5} w^j_{\text{rel}} \cdot Memb^j_{\text{out}} \tag{3.27}$$

3.2.4 Defuzzification

This is the inverse function of the recoding process. In fuzzy inductive reasoning, it is called *regeneration*. If the shape of the membership functions used in the recoding process is known, the regeneration of the quantitative (real–valued) data can be obtained in an unambiguous fashion, i.e., without loss of information. In this sense, the regeneration function is indeed the inverse of the recoding function. However, it is not being used to reconstruct the original data. Instead, it is being used to construct quantitative (real–valued) estimates of the forecasts made from the predicted qualitative triples.

In the following sections, specific aspects of the FIR methodology are introduced as they relate to the forecasting of time series in particular.

3.3 Characterizing Time Series

It had been pointed out already in Chapter 2 that, in time–series analysis, one or several outputs of a system can be observed, but the system cannot be excited through its inputs that generally are not even known. The objective of the time–series analysis usually is to predict the future behavior of a measured signal through observations of its behavior in the past. There exists a wide range of literature relating to different methodologies used for forecasting the behavior of time series (Weigend *et al.* 1990).

The oldest references discuss the use of the immediate previous values of a time series to predict their future behavior (Yule 1927). Soon, this idea was generalized to using a subset of the past history of a time series to predict its future behavior.

The knowledge available about a univariate time series can be represented by n values $\{x_1,\ x_2,\ \cdots, x_n\}$ observed in the past. In the process of observation, samples should either be drawn equidistantly (i.e., using a constant sampling rate), or alternatively, the time instants when the samples were taken need to be added as a tag (the series needs to be time–stamped). The prediction consists of looking for the future values $\{x_{n+1},\ x_{n+2},\ \cdots\}$ (Takens 1981).

If the time series is *deterministic* and *time–invariant*, there exists a scalar value d and a scalar function f, such that for each $t > d$:

$$x_t = f(x_{t-1}, x_{t-2}, \cdots, x_{t-d}) \tag{3.28}$$

The quantity d is often referred to as the *embedding dimension* of the time series. It corresponds to the number of *degrees of freedom* of the system from which the time series was generated. The function f, usually a highly non–linear relationship, characterizes the system that was used to produce the time series. Hence Equation (3.28) can be called a *model* of the system, simpler though less accurately, a generating "model" of the time series itself.

Given a set of n observations, the modeling task now is to find d and f. In the research presented in this dissertation, autocorrelation analysis was used to determine d, and FIR was used to determine f.

The literature on time–series analysis distinguishes between two different situations: the direct prediction and the interactive prediction. The *direct prediction* only uses real observations for the forecast, whereas the *interactive prediction* also uses previously made forecasts as if they were real observations. Sometimes, the former type is simply referred to as "prediction," whereas the latter kind is called "simulation."

3.4 Procedure for Multi–Step Prediction Analysis

Once the embedding dimension d and the functional relationship f have been identified (i.e., the *model* has been found), the following procedure is carried out as an attempt to standardize the multi–step prediction analysis.

Matrix (3.29) shows how the computations are performed. It shows an excerpt of the time series. The first column, with variables written in italic type, denotes the true measurement data. At each sampling point, a multi–step prediction is being performed, the results of which are written in roman type to the right of the last measurement data point used in the prediction. The first argument denotes the time instant for which the prediction is

computed, whereas the second argument denotes the number of prediction steps used to reach the prediction. Across one of the anti–diagonals, values are marked in bold type for illustration. They all refer to the same time point, yet values further to the right and top are less accurate, because they have been obtained using a longer prediction path (second argument).

$$Y = \begin{pmatrix} \dots & \dots & \dots & \dots & \dots & \dots \\ y(t-4\delta t) & \mathrm{y(t-3\delta t,1)} & \mathrm{y(t-2\delta t,2)} & \mathrm{y(t-\delta t,3)} & \mathrm{y(t,4)} & \dots \\ y(t-3\delta t) & \mathrm{y(t-2\delta t,1)} & \mathrm{y(t-\delta t,2)} & \mathrm{y(t,3)} & \mathrm{y(t+\delta t,4)} & \dots \\ y(t-2\delta t) & \mathrm{y(t-\delta t,1)} & \mathrm{y(t,2)} & \mathrm{y(t+\delta t,3)} & \mathrm{y(t+2\delta t,4)} & \dots \\ y(t-\delta t) & \mathrm{y(t,1)} & \mathrm{y(t+\delta t,2)} & \mathrm{y(t+2\delta t,3)} & \mathrm{y(t+3\delta t,4)} & \dots \\ y(t) & \mathrm{y(t+\delta t,1)} & \mathrm{y(t+2\delta t,2)} & \mathrm{y(t+3\delta t,3)} & \mathbf{y(t+4\delta t,4)} & \dots \\ y(t+\delta t) & \mathrm{y(t+2\delta t,1)} & \mathrm{y(t+3\delta t,2)} & \mathbf{y(t+4\delta t,3)} & \mathrm{y(t+5\delta t,4)} & \dots \\ y(t+2\delta t) & \mathrm{y(t+3\delta t,1)} & \mathbf{y(t+4\delta t,2)} & \mathrm{y(t+5\delta t,3)} & \mathrm{y(t+6\delta t,4)} & \dots \\ y(t+3\delta t) & \mathbf{y(t+4\delta t,1)} & \mathrm{y(t+5\delta t,2)} & \mathrm{y(t+6\delta t,3)} & \mathrm{y(t+7\delta t,4)} & \dots \\ \mathbf{y(t+4\delta t)} & \mathrm{y(t+5\delta t,1)} & \mathrm{y(t+6\delta t,2)} & \mathrm{y(t+7\delta t,3)} & \mathrm{y(t+8\delta t,4)} & \dots \\ y(t+5\delta t) & \mathrm{y(t+6\delta t,1)} & \mathrm{y(t+7\delta t,2)} & \mathrm{y(t+8\delta t,3)} & \mathrm{y(t+9\delta t,4)} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \tag{3.29}$$

The above matrix can now be used in different ways. Horizontal rows indicate individual multi–step predictions starting from the time shown in the first column that represents the last measurement data point. Each new data point contains more sources of error than the previous one, because it is built on a longer prediction history. Vertical columns show long–term prediction cycles, whereby the measurement data lack behind the prediction by a fixed number of steps. Columns further to the left should, on average, be more accurate than columns further to the right, because the prediction history leading to them is shorter. The first column is 100% accurate, since it represents the measurement data. Finally, values in anti–diagonals represent the same time instant estimated using longer and longer prediction histories.

As a standard experiment, multi–step predictions are being performed over 15 steps throughout most of the time series predicted in this dissertation. Errors between the true forecast and the predictions are computed along the anti–diagonals, and are averaged across columns.

3.5 The Prediction Error

There does not exist a single error formula that everyone would agree upon as being the best for evaluating success (or rather failure) of a prediction.

The *absolute error* of an i–step prediction, err_i, can be defined as:

$$err_i = y - \hat{y}_i \tag{3.30}$$

where $\hat{y}_i$ denotes the prediction of the observation y with the last observed value lagging i steps behind. Yet, such a measure would have the

disadvantage that the absolute values of the errors could not be compared across different time series.

Alternatively, one could use a variety of different formulae measuring the *relative error*, such as:

$$err_i = \frac{\text{abs}(y - \hat{y}_i)}{\text{mean}(y)} \tag{3.31}$$

which fails, if accidentally $\text{mean}(y) = 0.0$. Other formulae that are sometimes used include:

$$err_i = \frac{\text{abs}(y - \hat{y}_i)}{\max(\text{abs}(y), \text{abs}(\hat{y}_i), \epsilon)} \tag{3.32}$$

where ϵ is the smallest number that can be distinguished from 1.0 in addition (a predefined Matlab variable). This formula will never fail, but the addition of the fudge factor ϵ is not truly satisfactory in all situations, and it is not clear that this formula reflects well the intuitive understanding of "goodness of fit" that a human researcher would have when visually comparing a set of values $y(t)$ with their predictions $\hat{y}_i(t)$.

In this dissertation, another formula is being proposed that consists of four different components. The first component measures the accuracy with which the forecast predicts the *mean value* of the time series:

$$err_{\text{mean}_i} = \frac{\text{abs}(\text{mean}(y(t)) - \text{mean}(\hat{y}_i(t)))}{\max(\text{abs}(\text{mean}(y(t))), \text{abs}(\text{mean}(\hat{y}_i(t))), \epsilon)} \tag{3.33}$$

It is a relative error with a fudge factor. Yet, the fudge factor will never come to play, because it will only be applied when:

$$\text{mean}(y(t)) = \text{mean}(\hat{y}_i(t)) = 0.0 \tag{3.34}$$

in which case the numerator is exactly equal to zero.

The second component measures the accuracy, with which the *standard deviation* of the time series is being predicted:

$$err_{\text{std}_i} = \frac{\text{abs}(\text{std}(y(t)) - \text{std}(\hat{y}_i(t)))}{\max(\text{abs}(\text{std}(y(t))), \text{abs}(\text{std}(\hat{y}_i(t))), \epsilon)} \tag{3.35}$$

The same applies as above w.r.t. the fudge factor.

For the third and fourth component, the time series and its prediction are jointly normalized to the range $[0.0, 1.0]$. Let:

$$y_{\max} = \max(y(t), \hat{y}_i(t)) \tag{3.36}$$

where the *max*–operator is applied to the concatenated series consisting of $y(t)$ and $\hat{y}_i(t)$, and similarly:

$$y_{\min} = \min(y(t), \hat{y}_i(t)) \tag{3.37}$$

Normalized time series can be computed as:

$$y_{\text{norm}}(t) = \frac{y(t) - y_{\min}}{\max(y_{\max} - y_{\min}, \epsilon)} \tag{3.38}$$

and similarly:

$$y_{\text{norm}_i}(t) = \frac{\hat{y}_i(t) - y_{\min}}{\max(y_{\max} - y_{\min}, \epsilon)} \tag{3.39}$$

Since the two curves have been normalized, it is now possible to use the absolute errors. The pointwise absolute error between the two curves $y(t)$ and $\hat{y}_i(t)$ can be computed as:

$$err_{\text{abs}_i}(t) = \text{abs}(y_{\text{norm}}(t) - y_{\text{norm}_i}(t)) \tag{3.40}$$

It is also possible to define the pointwise *similarity* between the two curves as:

$$sim_i(t) = \frac{\min(y_{\text{norm}}(t), y_{\text{norm}_i}(t))}{\max(y_{\text{norm}}(t), y_{\text{norm}_i}(t), \epsilon)} \tag{3.41}$$

where, this time around, the *min*– and *max*–operators are being applied elementwise rather than to the concatenated time series.

Using the pointwise similarity, a pointwise *similarity error* can be defined as:

$$err_{\text{sim}_i}(t) = 1.0 - sim_i(t) \tag{3.42}$$

The averaged absolute and similarity error is then defined as:

$$err_{\text{avg}_i} = \text{mean}(err_{\text{abs}_i}(t) + err_{\text{sim}_i}(t)) \tag{3.43}$$

Finally, the total error, in percentage, is the sum of the four components multiplied by 25.0:

$$err_{\text{tot}_i} = 25.0 \cdot (err_{\text{mean}_i} + err_{\text{std}_i} + err_{\text{avg}_i}) \tag{3.44}$$

The factor 25.0 is justified, because there are four separate components that all measure different aspects of one and the same thing. Each one of them is usually in the range $[0.0, 1.0]$ (although some errors could be larger than

1.0 at times), thus, the accumulated total error should be somewhere in the range [0%,100%] most of the time.

The proposed formula is a compromise that was developed over a long period of time, and that saw many revisions over the years. It results in a quantification of success (or rather failure) of predictions that is, as shall be seen, quite consistent with the intuitive understanding of success (failure) that a human observer would have when comparing $y(t)$ and $\hat{y}_i(t)$ by the naked eye.

3.6 Two Examples

In order to explain how the FIR methodology is being applied to time–series analysis, this section introduces two time series that have quite different characteristics, one representing a single mode far infrared NH_3 laser beam, the other representing the number of cases of the meningitis disease in Barcelona and its suburban regions. In order to distinguish easily between the different series that shall be introduced in this dissertation, each is identified by a single uppercase character. The laser series shall henceforth be called *Series L*, and the meningitis series shall be called *Series M*.

3.6.1 Forecasting Time Series L

Figure 3.4 shows the chaotic intensity pulsations of a single–mode far infrared NH_3 laser beam. This time series was first presented in (Weigend and Gershenfeld 1994). 9800 data points are available. The first 1000 points were used for *training*, i.e., to construct the experience data base. The sequence starting from sample 8601 and ending with sample 9800 was used for *testing*, i.e., to verify the success (failure) of the prediction. The testing sequence contains three amplitude switch–overs. It will be of particular interest to observe how well FIR performs in predicting these switch–over events. Multi–step predictions were performed over 15 samples.

Table 3.1 characterizes (classifies) Series L using the nomenclature introduced in Chapter 2.

The behavior of this time series is highly regular, though chaotic. Although it may be difficult to predict, with high precision, the amplitude of each peak and the time of the next amplitude switch–over event, the behavior in between peaks should be well predictable. The time series does not look stochastic at all, i.e., the "signal–to–noise" ratio is high.

Figure 3.5 shows the autocorrelation of the training data.

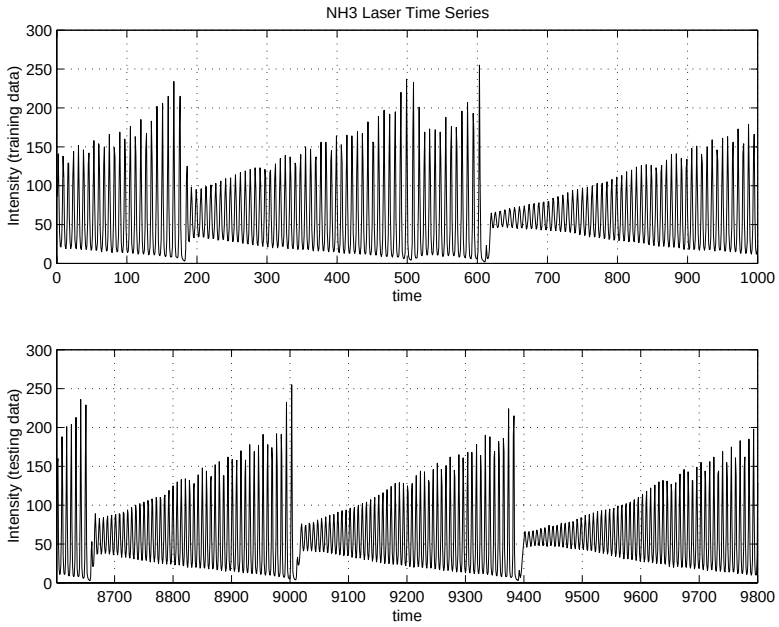

Figure 3.4: Chaotic intensity pulsations in a single–mode far infrared NH_3 laser. For training, the first 1000 data points were used, whereas data points 8601 to 9800 served for testing

It can be seen quite clearly that there is a strong correlation over 7 samples. Since also data point 8 still has a very high auto–correlation, it was decided to choose the embedding dimension to be $d = 8$. Consequently, the mask candidate matrix takes the form:

$$mcan = \begin{array}{c} {}_t\backslash^x \\ t-8\delta t \\ t-7\delta t \\ t-6\delta t \\ t-5\delta t \\ t-4\delta t \\ t-3\delta t \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y_1 \\ \left(\begin{array}{c} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ +1 \end{array} \right) \end{array} \tag{3.45}$$

The data were recoded into three levels. An *optimal mask analysis* was then performed in SAPS–II (our current implementation of the FIR methodology). It resulted in the following optimal mask:

Table 3.1: Classification of Time Series L.

natural	L	synthetic	
stationary	L	non–stationary	
time invariant	L	time varying	
low dimensional	L	stochastic	
clean	L	noisy	
short		long	L
dormant		active	L
documented	L	blind	
linear		non–linear	L
scalar	L	vector	
single recording	L	multiple recordings	
continuous	L	discrete	

$$mask = \begin{array}{c} {}_t\backslash^x \\ t-8\delta t \\ t-7\delta t \\ t-6\delta t \\ t-5\delta t \\ t-4\delta t \\ t-3\delta t \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y_1 \\ \left(\begin{array}{r} 0 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -2 \\ +1 \end{array} \right) \end{array} \tag{3.46}$$

FIR decided that it needed a mask of complexity 3 with a depth of 8 ($d = 7$). It decided further that the best prediction can be obtained by looking at the immediate past value and also the value 7 samples ago. This is certainly reasonable. FIR determined that the quality of this mask is $Q_3 = 0.6977$.

FIR offered the following alternative "good" masks:

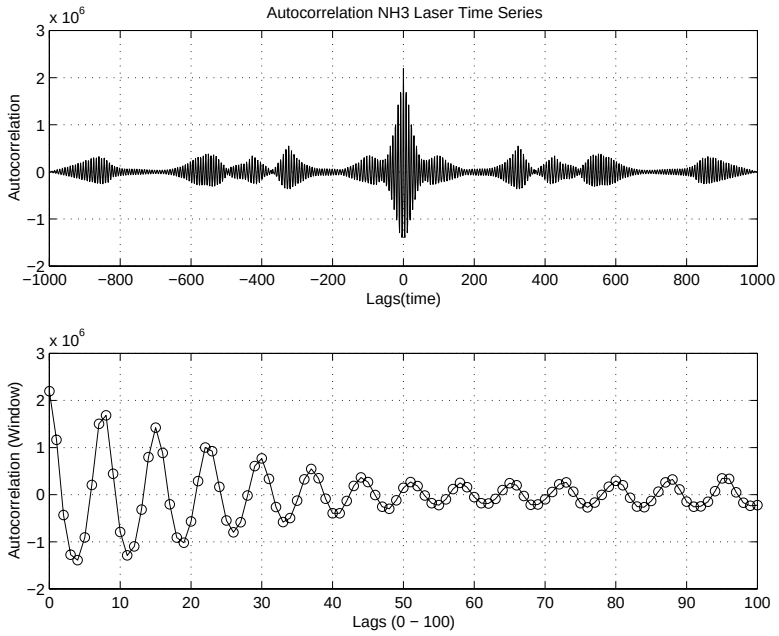

Figure 3.5: Auto–correlation of the training data of Series L.

$$
\begin{array}{c|c}
{}_t\backslash^x & y_1 \\
t-8\delta t & 0 \\
t-7\delta t & -1 \\
t-6\delta t & 0 \\
t-5\delta t & 0 \\
t-4\delta t & 0 \\
t-3\delta t & 0 \\
t-2\delta t & 0 \\
t-\delta t & 0 \\
t & +1
\end{array}
\qquad
\begin{array}{c|c}
{}_t\backslash^x & y_1 \\
t-8\delta t & 0 \\
t-7\delta t & -1 \\
t-6\delta t & 0 \\
t-5\delta t & 0 \\
t-4\delta t & 0 \\
t-3\delta t & -2 \\
t-2\delta t & 0 \\
t-\delta t & -3 \\
t & +1
\end{array}
\qquad
\begin{array}{c|c}
{}_t\backslash^x & y_1 \\
t-8\delta t & 0 \\
t-7\delta t & -1 \\
t-6\delta t & 0 \\
t-5\delta t & -2 \\
t-4\delta t & 0 \\
t-3\delta t & -3 \\
t-2\delta t & 0 \\
t-\delta t & -4 \\
t & +1
\end{array}
\tag{3.47}
$$

of complexities 2, 4, and 5, respectively. The corresponding mask qualities were: $Q_2 = 0.4857$, $Q_4 = 0.6182$, and $Q_5 = 0.3615$.

The mask of complexity 2 is probably not reasonable, because it does not make use of the immediate past value, i.e., it will not recognize switch–over events for a long time. The mask of complexity 4 is quite reasonable. It might be a good alternative to that of complexity 3. The mask of complexity 5 cannot be justified on the basis of the available training data. Data deprivation has set in, and has drastically reduced the quality of the mask (because of a low OR value). More training data would be needed to justify a mask of such high complexity.

Figure 3.6 shows the average total error as a function of the number of samples to be predicted. Clearly, for 0 samples of prediction, the error must be zero, because the prediction simply is the real data stream. A 1–sample forecast is a "prediction" in the true sense, as it only uses real data for making the prediction. All longer–term forecasts are "simulations," as they make use of previously made predictions in determining the forecast.

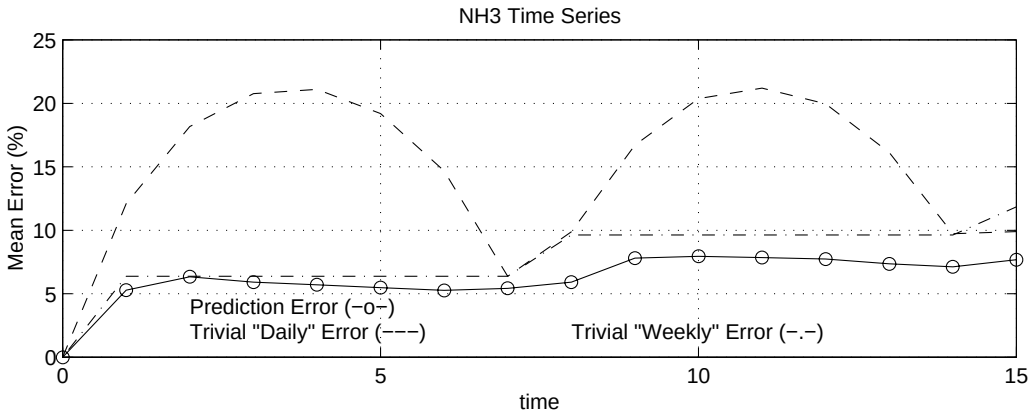

Figure 3.6: Comparative error analysis of prediction

The dashed curve is the *trivial "daily" prediction*, which was added for comparison. The trivial "daily" prediction simply forecasts that "today's" value is the same as "yesterday's". The dot–dashed curve is the *trivial "weekly" prediction*. It forecasts that "today's" value is the same as that of "one week ago[1]."

Clearly, FIR needs to do better than both the daily and weekly trivial predictions in order to claim that it has accomplished anything of significance.

FIR predicts considerably better than the trivial one–sample ("daily") prediction. However, because of the strong auto–correlation for 7 samples, the error of the trivial prediction is much smaller for a 7 sample delay than for a smaller delay. This fact can be exploited, by voluntarily increasing the delay to multiples of 7 samples at all times, i.e., the predictions up to 7 samples ahead are based on observations 7 samples earlier, whereas the predictions over 8 to 14 samples are based on observations that lag 14 samples behind, etc. This is the so–called "trivial weekly prediction."

FIR also outperforms the trivial "weekly" prediction, but not by as much as one might have hoped. It is this gain in performance over the trivial prediction that can be considered FIR's "work."

[1]The terms "daily" and "weekly" are used here in a metaphorical sense.

Figure 3.7 shows the original data, the 1–step forecast, the 8–step forecast, and the 15–step forecast. Figure 3.8 shows an excerpt of the same forecasts around the sample 9000, i.e., during an amplitude switch–over event.

Figure 3.7 shows that FIR essentially was able to learn the behavior of this time series. Although the series is only stationary over a long time frame (much longer than the depth of the mask), FIR reproduces quite well the amplitude growth patterns, and it is also capable of dealing well with the amplitude switch–over events.

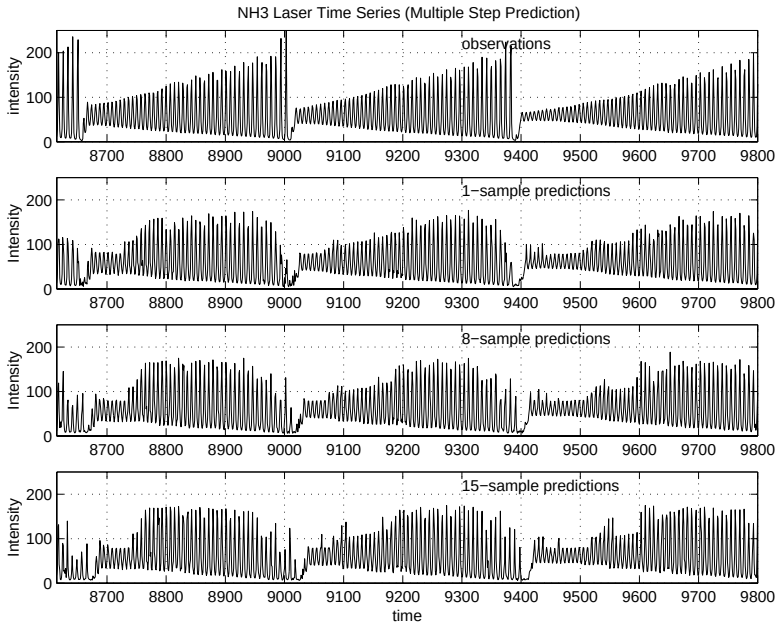

Figure 3.7: Prediction and simulation results of Time Series L

Figure 3.8 shows an excerpt of Figure 3.7 around such a switch–over event. The delay in recognizing the event (which essentially cannot be predicted until it occurs) is exactly equal to the number of samples that the real data lag behind. Yet, once the event has been recognized, FIR deals with it confidently and reliably.

It would have been possible to obtain yet better results by trying different masks or choosing a different testing window, but this is beside the point. The purpose is not to show beautiful results, but to learn something about FIR's capabilities, and demonstrate what FIR can do on its own. Thus, the results were accepted as they were produced by FIR on its first run, without ever trying to "manicure" the results in any way.

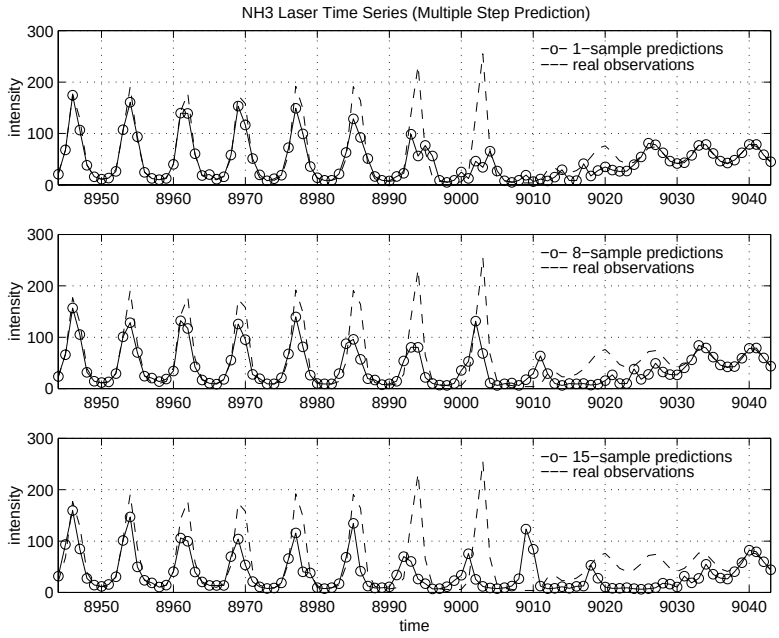

Figure 3.8: Window of the simulation results on Time Series L.

3.6.2 Forecasting Time Series M

Figure 3.9 shows the number of cases of the meningitis disease in the city of Barcelona and its surroundings. Monthly data are available starting from January 1963 until December 1996. Hence 400 data points are available. The first 350 points were used for *training*. The remaining 50 samples were used for testing.

Series M can be characterized as shown in Table 3.2 using the nomenclature introduced in Chapter 2.

The behavior of this time series is highly stochastic. Looking at Figure 3.9, one might conclude that Series M is stationary over a long time period. However, there are not enough data available to truly support such a statement. Moreover, the notion may even be incorrect. Over such a long time frame (the data reach over more than 30 years, the population density in the Barcelona region, the frequency of travels across region boundaries, and the medical support system, three important factors determining the epidemiological dynamics of the disease, have certainly not remained the same. It will thus be quite interesting to see how FIR reacts to such a time series, whether it would predict anything reasonable, and, even more interestingly, whether it is aware that it might be trying to predict unpredictable data.

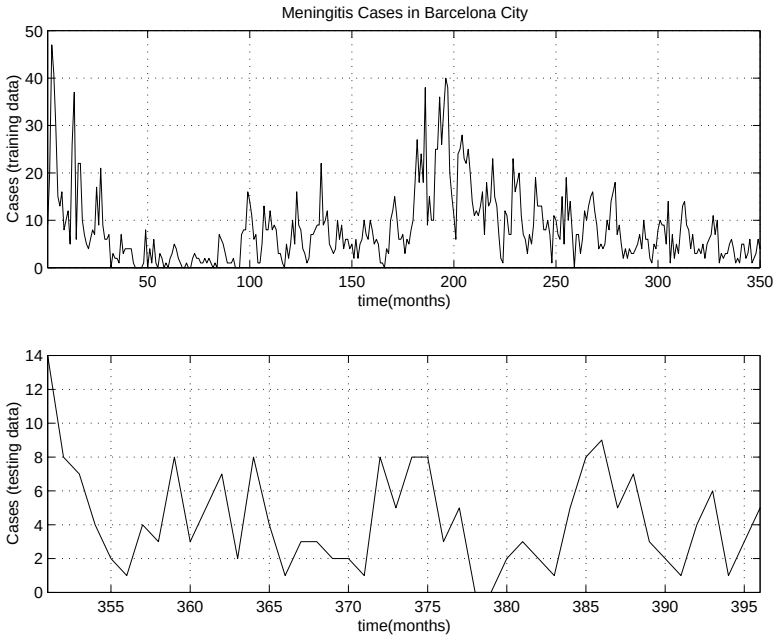

Figure 3.9: Barcelona Meningitis Cases. 350 monthly samples were used for training, the remaining 50 samples were used for testing. Data are available starting from January of 1963, and ending with December of 1996.

Figure 3.10 shows the autocorrelation of the training data.

It can be seen quite clearly that there exists an seasonal correlation. However, the correlation is not very strong. For this reason, it was decided to give FIR a little more than a year to work with. The embedding dimension was thus chosen to be $d = 15$. Consequently, the mask candidate matrix takes the form:

$$mcan = \begin{array}{c} {}_t\backslash^x \\ t-15\delta t \\ t-14\delta t \\ \vdots \\ \vdots \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y_1 \\ \begin{pmatrix} -1 \\ -1 \\ \vdots \\ \vdots \\ -1 \\ -1 \\ +1 \end{pmatrix} \end{array} \tag{3.48}$$

As in the previous case, the data were recoded into three levels. The optimal mask analysis resulted in the following model:

Table 3.2: Classification of Time Series M

natural	M	synthetic	
stationary		non–stationary	M
time invariant	M	time varying	
low dimensional		stochastic	M
clean		noisy	M
short	M	long	
dormant		active	M
documented	M	blind	
linear		non–linear	M
scalar	M	vector	
single recording	M	multiple recordings	
continuous		discrete	M

$$mask = \begin{array}{c} {}_t\backslash^x \\ t-15\delta t \\ t-14\delta t \\ t-13\delta t \\ t-12\delta t \\ t-11\delta t \\ t-10\delta t \\ t-9\delta t \\ \vdots \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y_1 \\ \left(\begin{array}{r} 0 \\ -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ \vdots \\ 0 \\ -3 \\ +1 \end{array} \right. \end{array} \tag{3.49}$$

FIR decided that the optimal model consists of a mask of complexity 4 with a depth of 15 ($d = 14$). It decided further that the best prediction can be obtained by looking at the immediate past value and also the values 10, and 14 samples ago. Thus, although the auto–correlation function has a relative maxima at 11 samples ago, FIR decided that a different selection of m–inputs provides better forecasts. It determined the quality of the optimal mask to be $Q_4 = 0.3485$.

FIR offered the following alternative "good" masks:

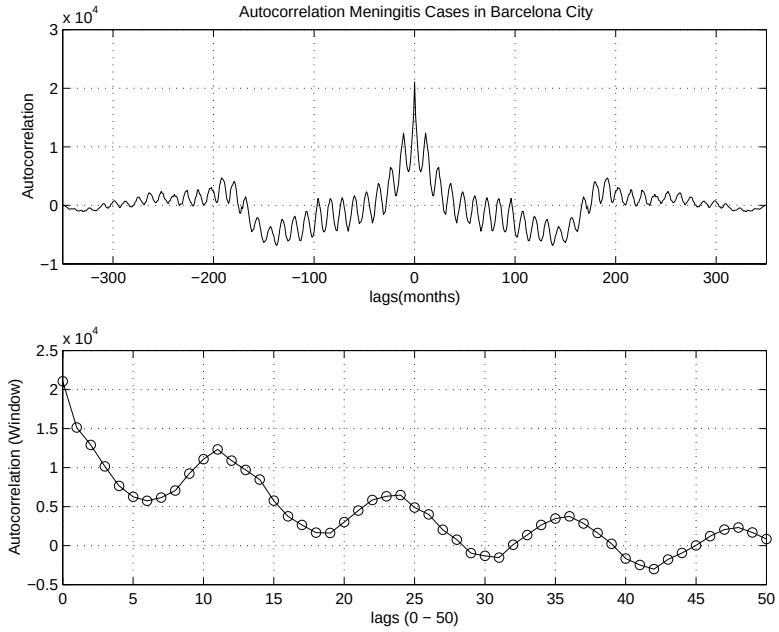

Figure 3.10: Auto–correlation of the training data of Series M.

$$
\begin{array}{c|c}
{}_t\backslash^x & y_1 \\
t-15\delta t & 0 \\
t-14\delta t & 0 \\
t-13\delta t & 0 \\
t-12\delta t & 0 \\
t-11\delta t & 0 \\
t-10\delta t & 0 \\
t-9\delta t & 0 \\
t-8\delta t & 0 \\
t-7\delta t & 0 \\
t-6\delta t & 0 \\
t-5\delta t & 0 \\
t-4\delta t & 0 \\
t-3\delta t & 0 \\
t-2\delta t & 0 \\
t-\delta t & -1 \\
t & +1
\end{array}
\qquad
\begin{array}{c|c}
{}_t\backslash^x & y_1 \\
t-15\delta t & 0 \\
t-14\delta t & 0 \\
t-13\delta t & 0 \\
t-12\delta t & 0 \\
t-11\delta t & -1 \\
t-10\delta t & 0 \\
t-9\delta t & 0 \\
t-8\delta t & 0 \\
t-7\delta t & 0 \\
t-6\delta t & 0 \\
t-5\delta t & 0 \\
t-4\delta t & 0 \\
t-3\delta t & 0 \\
t-2\delta t & 0 \\
t-\delta t & -2 \\
t & +1
\end{array}
\qquad
\begin{array}{c|c}
{}_t\backslash^x & y_1 \\
t-15\delta t & 0 \\
t-14\delta t & -1 \\
t-13\delta t & 0 \\
t-12\delta t & 0 \\
t-11\delta t & 0 \\
t-10\delta t & -2 \\
t-9\delta t & 0 \\
t-8\delta t & 0 \\
t-7\delta t & 0 \\
t-6\delta t & -3 \\
t-5\delta t & 0 \\
t-4\delta t & 0 \\
t-3\delta t & 0 \\
t-2\delta t & 0 \\
t-\delta t & -4 \\
t & +1
\end{array}
\tag{3.50}
$$

of complexities 2, 3, and 5, respectively. The corresponding mask qualities were: $Q_2 = 0.2532$, $Q_3 = 0.3169$, and $Q_5 = 0.3011$.

In general, the quality values are lower than those obtained for Series L. All proposed masks are quite reasonable. The auto–correlation cycles are not strong enough to lock FIR into any particular patterns. FIR decides that it is best to distribute the m–inputs more or less regularly over the available time window, except with the immediate past sample that is selected always.

Figure 3.11 shows the average total error as a function of the number of samples to be predicted.

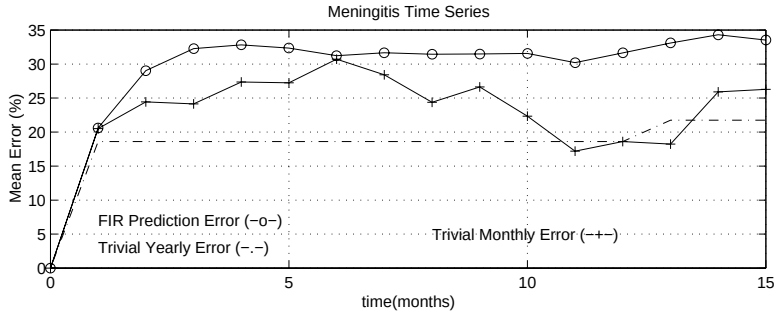

Figure 3.11: Comparative prediction error analysis of Series M.

The curve marked with '+' symbols is the *trivial monthly prediction*, which was added for comparison. The trivial monthly prediction forecasts that today's number of meningitis cases is the same as that of one month ago. The dot–dashed curve is the *trivial annual prediction*. It forecasts that today's value is the same as that of one year ago.

It is quite evident that, this time, FIR does not accomplish anything. Both of the trivial predictions outperform FIR by leaps and bounds.

Figure 3.12 shows the original data, the 1–month forecast, the 8–month forecast, and the 15–month forecast.

FIR does not predict impossible or even improbable outcomes. It would not know how to, as it can only predict patterns that it has observed before. The 1–month predictions look somehow similar to the real observations, though by no means better than the trivial prediction that would simply lag one month behind the real observations. The 15–month prediction still estimates the mean value correctly, but does not accomplish much more than that.

It was mentioned in Chapter 1 that one of FIR's foremost advantages is its capability to recognize its own mistakes. How come that FIR still was able to make predictions that are essentially garbage? The answer is that up to this point, the author chose to ignore FIR's warning messages. Figure 3.13 shows the average *accumulated confidence* that FIR has in its own predictions over 15 steps for the two time series. Chapter 5 of this thesis will explain

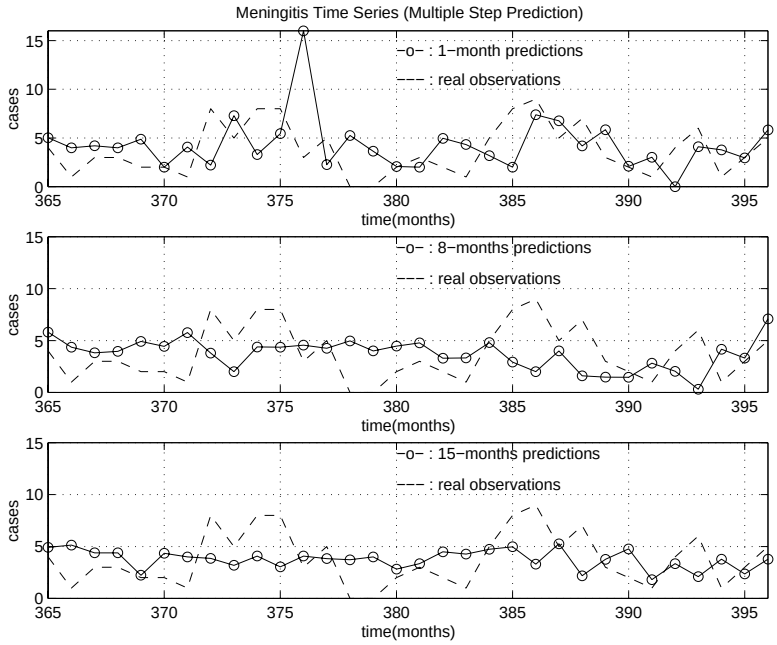

Figure 3.12: FIR prediction and simulation of Series M.

how FIR goes about to compute the confidence value.

Figure 3.13 shows that, for Series L, FIR has an average confidence of 87% during its first prediction step. In contrast, the average confidence for Series M is only 50% during the first step, i.e., FIR tells the user that, with 50% likelihood, already the first prediction is garbage. The user is well advised to heed this warning, and not place too much trust on the results obtained.

3.7 Forecasting Noise: Time Series N

At this point, it was decided to introduce yet another "time series." This series consists of uniformly distributed noise in the range $[0.0, 1.0]$, generated by Matlab's random number generator. Clearly, FIR (or any other technique for that matter) cannot be expected to outperform the trivial predictor on this series.

Figure 3.14 shows the average errors and the accumulated confidence for Series N. FIR's error is compared with the one–sample trivial prediction on the one hand, and with the *naïve prediction* on the other. The naïve prediction consists of predicting the constant value of 0.5.

Since the naïve predictor does not predict the standard deviation well, it

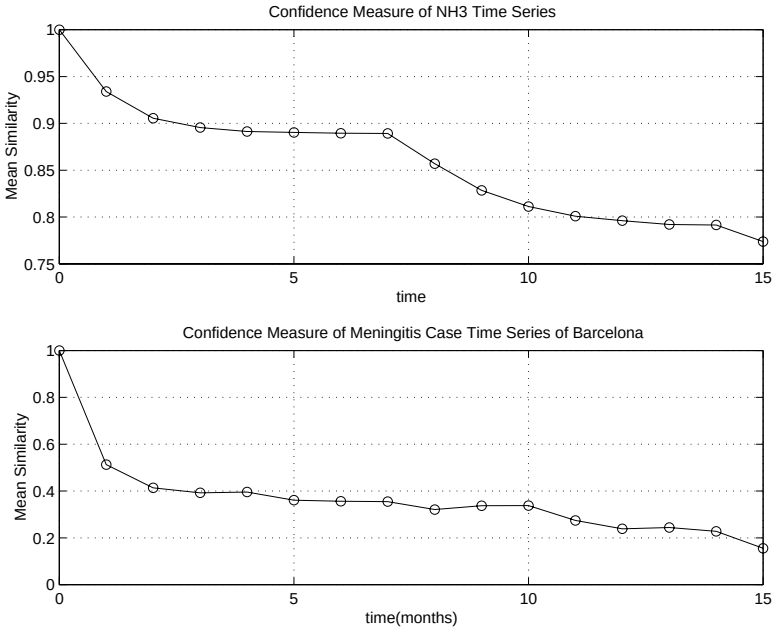

Figure 3.13: Accumulated FIR confidence of Series L and M.

is severely punished using the proposed error formula. The trivial predictor always exhibits zero error for both the mean and the standard deviation, which gives it quite an advantage. FIR performs somewhere in the middle between the trivial and the naïve predictor, but, for larger prediction periods, approaches more and more the performance of the naïve predictor. As in the case of Series M, the confidence in the validity of its predictions decreases rapidly.

Figure 3.15 shows the auto–correlation of the original "time series" and of the 1–sample, 8–sample, and 15–sample predictions. There is no significant auto–correlation in either case, i.e., FIR predicts indeed noise, as it should.

Figure 3.16 shows histograms of the original data as well as of the 1–sample, 8–sample, and 15–sample predictions. Whereas the original data are uniformly distributed, the FIR predictions resemble more normal distributions. The standard deviations decrease with increasing prediction time.

These results can be easily explained. Since the samples themselves are random, also the fuzzy membership function values are random. They are no longer uniformly distributed because of the non–linear map, but they are certainly random in the range $[0.5, 1.0]$. The 5–nearest neighbor method calculates the fuzzy membership value of the output as the mean of five such random samples. Because of the law of great numbers, this mean will

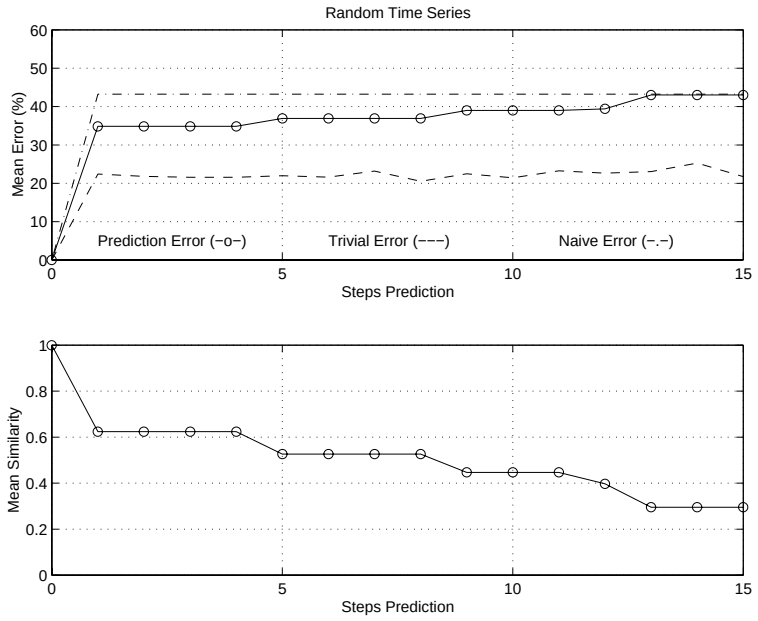

Figure 3.14: Average errors and accumulated confidence for Series N.

be almost normally distributed, and will have a reduced standard deviation because of the effect of averaging.

FIR, quite evidently, filters out noise. This can be seen in Figure 3.17 that shows the 1–sample, 8–sample, and 15–sample predictions.

The 15–sample prediction is almost a straight line. FIR has successfully filtered out all the noise, and only predicts the mean value, i.e., behaves just like the naïve predictor. Is this bad? There is no easy answer! If Series N is considered to represent a meaningful signal, then the best prediction would indeed be the trivial prediction. However, if Series N is considered noise, i.e., the only signal underneath the noise is the mean value of 0.5, then the naïve prediction is indeed the best there can be.

How does FIR decide what is "signal" and what is "noise"? If a (deterministic) pattern has been seen before, then FIR will find good neighbors, and will predict what it has seen. In contrast, if there is no deterministic pattern, or if the pattern is superimposed with noise, then FIR will not find good neighbors, and the effects of averaging will set in, with the consequence that the non–repetitive aspects of the signal will eventually get filtered out.

How well does FIR exploit the information provided to it, i.e., when does it consider a signal a signal, and when will it treat it as noise? The next section will shed some light on this question.

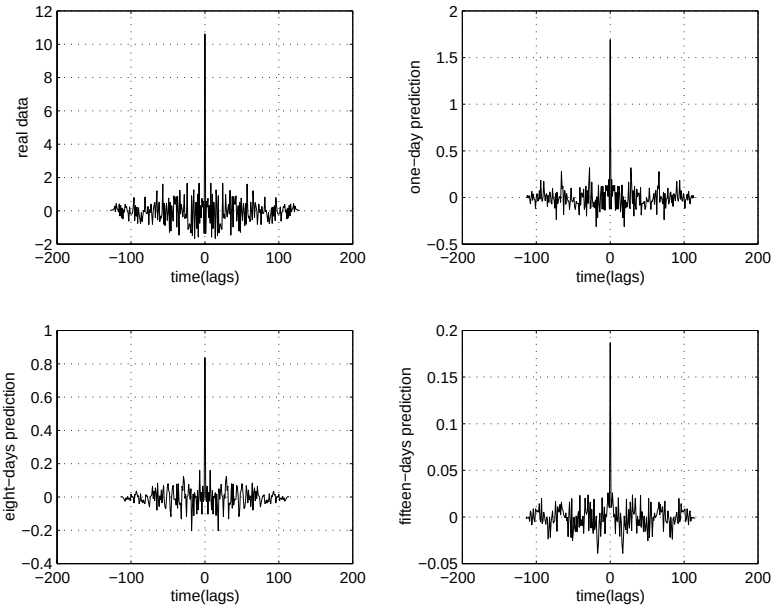

Figure 3.15: Comparative auto-correlation analysis of Series N.

3.8 Adding More Information: Time Series I

At this point, it is useful to introduce yet another time series. This time series is artificially constructed from Series L, by subtracting its mean value, and then integrating the time series over time. Series I is depicted in Figure 3.18.

It was necessary to subtract the mean before integration in order to ensure that also Series I is stationary over a sufficiently large time frame.

Using exactly the same methodology as earlier, FIR finds the very same optimal mask:

$$mask = \begin{array}{c} {}_t\backslash^x \\ t-7\delta t \\ t-6\delta t \\ t-5\delta t \\ t-4\delta t \\ t-3\delta t \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y_1 \\ \left(\begin{array}{r} -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -2 \\ +1 \end{array}\right) \end{array} \tag{3.51}$$

The averaged errors over a 15–sample prediction are shown in Figure 3.19.

The errors are almost identical to those found for Series L. The confidence

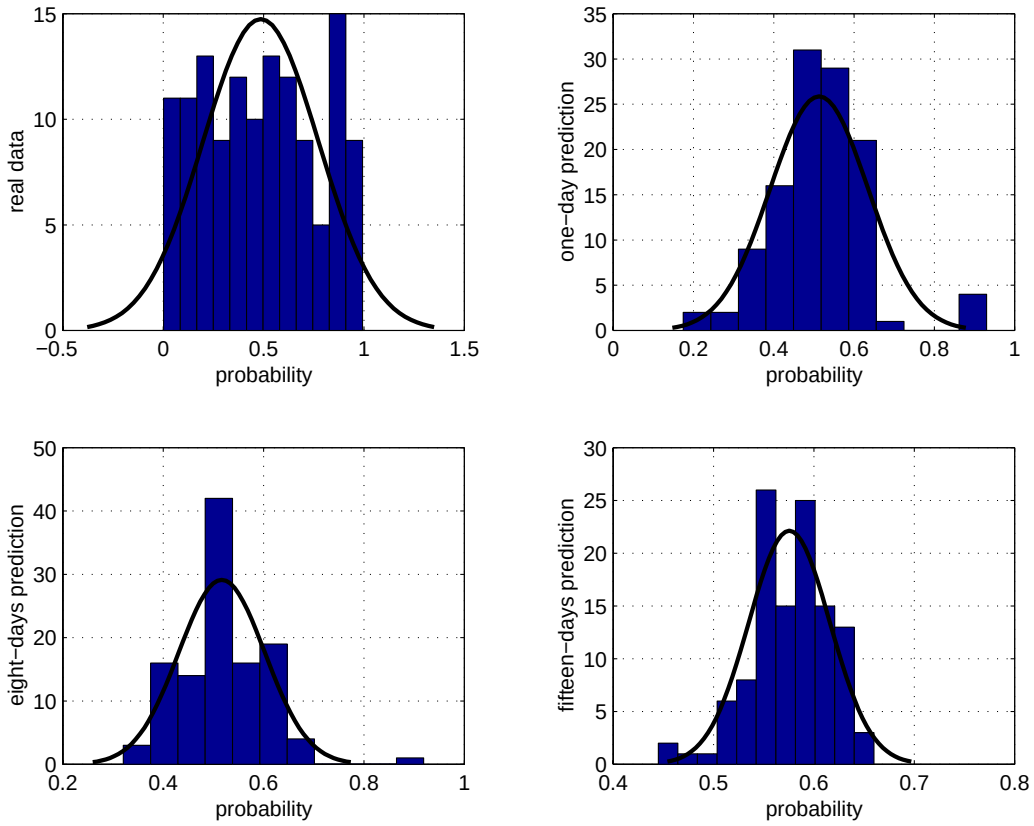

Figure 3.16: Comparative histogram analysis of Series N.

values are somewhat higher, because it was decided to use 4500 values for model identification instead of 1000, and therefore, FIR finds neighbors that are closer to the data point that it tries to predict, which increases its confidence in its prediction (cf. Chapter 5).

Figure 3.20 shows the 1–sample, 8–sample, and 15–sample predictions over the testing window. As before, the delay in recognizing the switch–over event is quite evident.

It shall now be investigated, how well FIR makes use of the available information. Since the system from which the time series was generated is obviously a higher–order system, it seems reasonable to assume that FIR would predict the output better, if it would have access to additional state information.

This is why this time series was constructed using an integrator. The derivative is available analytically, and it might make sense to offer this derivative as additional measurement data, i.e., treat the two signals together as a multivariate time series.

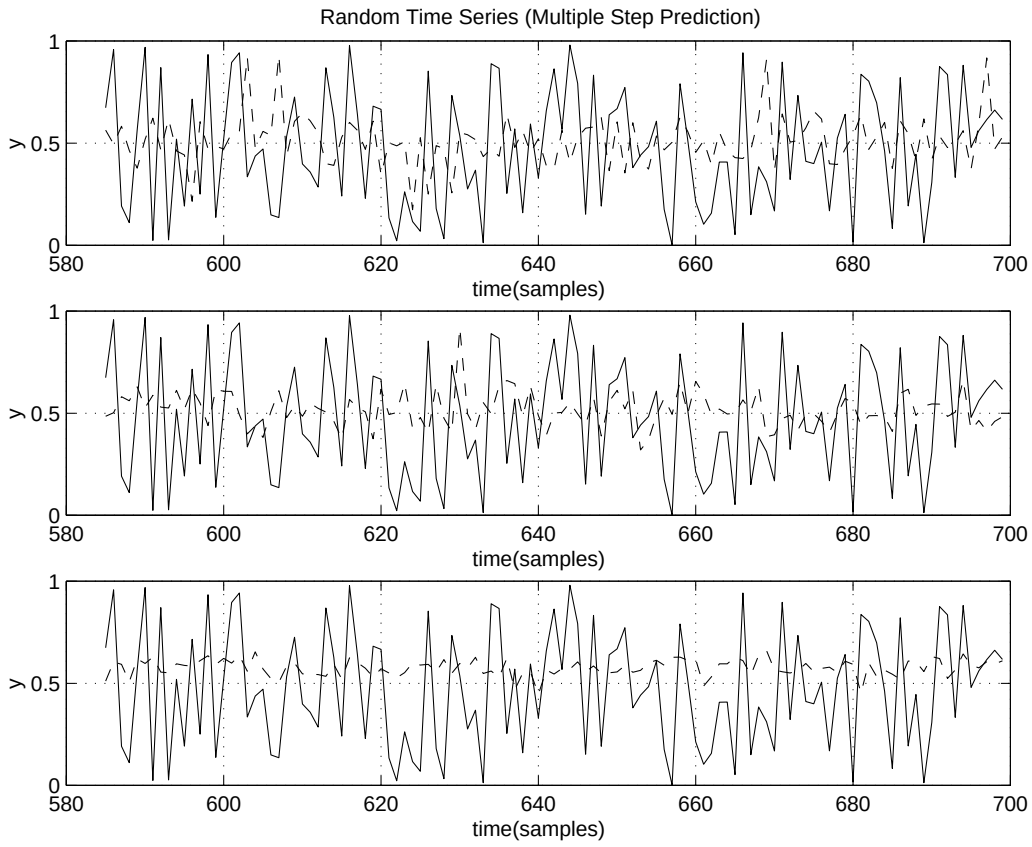

Figure 3.17: Simulation results for one-step and multiple-step predictions of Series N.

The following masks were used for the prediction:

$$mask_1 = \begin{array}{c} {}_t\backslash^x \\ t-7\delta t \\ t-6\delta t \\ t-5\delta t \\ t-4\delta t \\ t-3\delta t \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y_1 \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{pmatrix} -1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -2 & -3 \\ +1 & 0 \end{pmatrix} \qquad mask_2 = \begin{array}{c} {}_t\backslash^x \\ t-7\delta t \\ t-6\delta t \\ t-5\delta t \\ t-4\delta t \\ t-3\delta t \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y_1 \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{pmatrix} 0 & -1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -2 & -3 \\ 0 & +1 \end{pmatrix} \tag{3.52}$$

where y_1 stands for the output of Series I, and y_2 is its derivative.

Figure 3.21 shows the averaged errors over a 15–sample prediction.

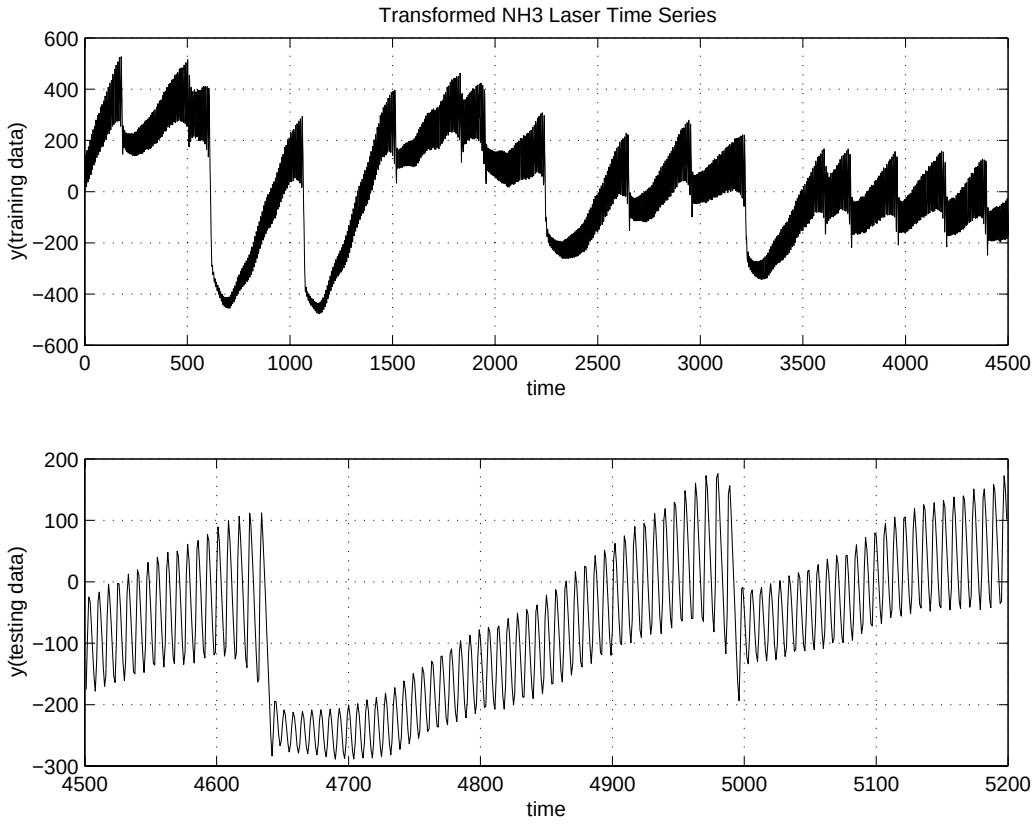

Figure 3.18: Integrated NH_3 Laser Time Series (Series I).

For 2, 8, and 9 samples, the errors are indeed smaller. The additional state information helped FIR improve its forecast. For other lag values, the results are not as favorable. The confidence values are lower again, because the mask now makes use of the additional state information, and the complexity of the mask is higher. Therefore, FIR no longer finds neighbors in the input space that are as close as they had been before, which reduces its confidence (cf. Chapter 5).

It is now interesting to compare the previous case with that where the quantitative (analytical) derivatives have been replaced by qualitative (numerical) derivatives. These can be obtained simply by subtracting neighboring samples using the DIFF–operator of SAPS–II. Figure 3.22 shows the averaged errors over a 15–sample prediction.

The results look now exactly as they had looked originally, i.e., nothing was gained by adding qualitative derivatives. The reason is that no additional information has been added. The same information is simply presented in another form. Since FIR evidently is quite capable of extracting *all* of the

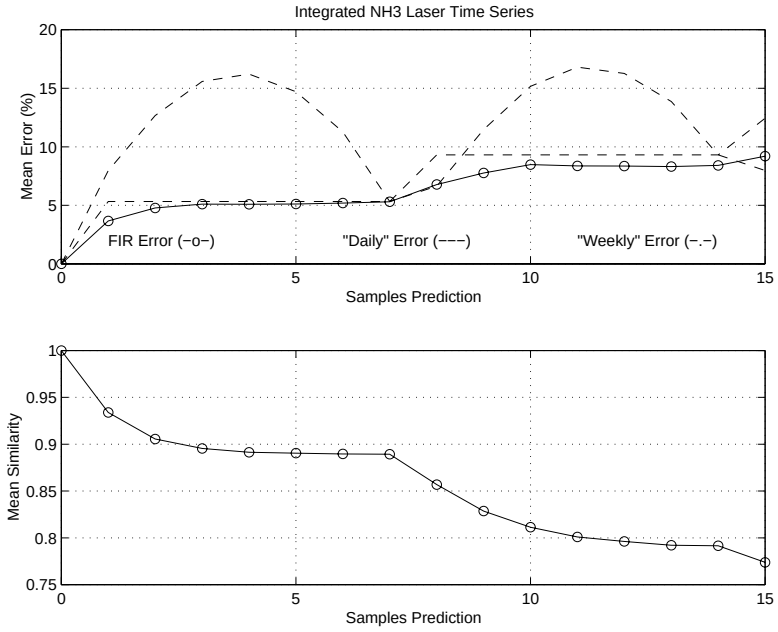

Figure 3.19: Mean prediction error and accumulated confidence of Series I.

information that is given to it, adding another representation of the same information does not help.

It might be useful to represent yet another experiment, one that is quite a bit more costly to execute. This time, 3000 samples shall be used for training the original model, and a prediction will be made over another 3000 samples.

Let $y(t)$ denote the observations, and $\hat{y}(t)$ represent the predictions. It is possible to make a model of the error:

$$e(t) = y(t) - \hat{y}(t) \tag{3.53}$$

It is now possible to use 2900 of the 3000 data points of the $e(t)$ trajectory to make a *model of the error*, $\hat{e}(t)$.

If $e(t)$ were known, it would be possible to reconstruct the observations without error:

$$y(t) = \hat{y}(t) + e(t) \tag{3.54}$$

Since there now exists a model of the error, one could try to get an improved prediction using:

$$\tilde{y}(t) = \hat{y}(t) + \hat{e}(t) \tag{3.55}$$

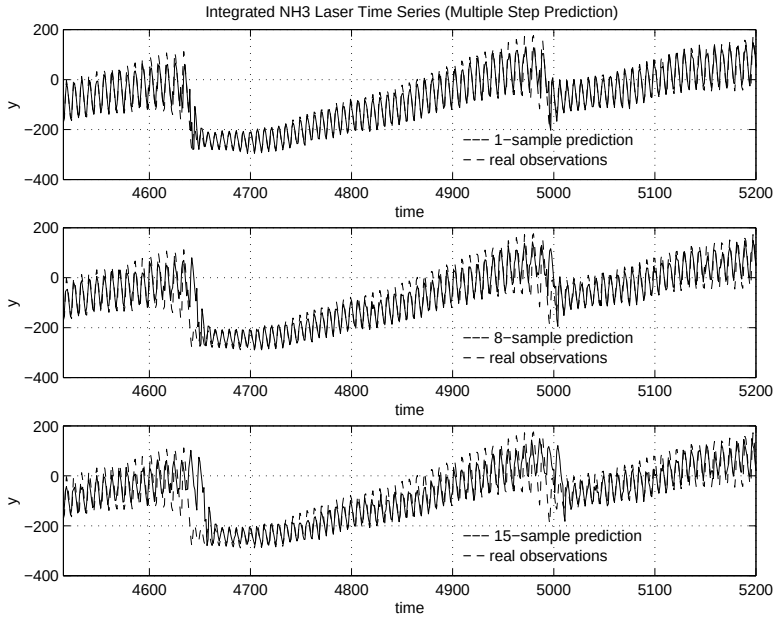

Figure 3.20: Simulation results for one-step and multiple-step predictions of Series I.

Figure 3.23 shows the averaged error of three models: the original model of the output: $\hat{y}$, an improved model using the error model as a second input variable (same idea as with the derivative variable), and finally, the superposition model: $\tilde{y}(t)$.

Evidently, the approach did not work. The original model is best except for one data point, where the superposition model gave a slightly smaller error. Figure 3.24 compares the 1–sample predictions of the three models.

Using the error variable as a second input, was a bad idea. It only prevented FIR from finding good neighbors, and distracted it from doing its job. It needs to be said that FIR did not propose the "optimal" mask that was used to compute the output variable. FIR proposed exactly the same optimal mask that was used in the original attempt, i.e., proposed to ignore the error variable entirely in the prediction of the output. The author decided to add the error variable into the mask to make the results look "more interesting."

The superposition idea did improve the forecast slightly. Figure 3.25 shows the averaged errors of the error model, i.e., the errors of the model that computes the error.

The errors are very large. The error model is difficult to predict, although the error trajectory is not free of auto–correlation. Figure 3.26 shows the 1–

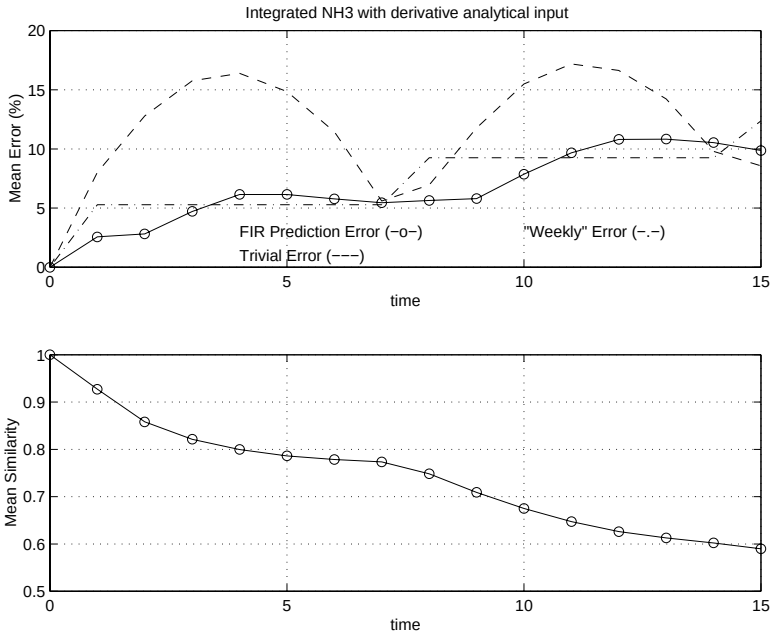

Figure 3.21: Average error and accumulated confidence of Series I using the analytical derivative as a second input.

sample, 8–sample, and 15–sample predictions of the error. The 1–sample prediction is decent. The 15–sample prediction is essentially a straight line. Once again, FIR has concluded that the data are mostly noise, and has filtered the noise out, keeping only the mean value, which is 0.0 in this case.

The error model contains *some* information that should be exploitable. Yet, the errors are small in comparison with the real output, and the errors of the errors are comparatively large. Therefore, the information contained in the error model gets lost among the noise in the superposition.

It can be concluded that FIR indeed exploits essentially *all* of the information that it is provided with. It does so confidently and reliably. No special tricks are needed to make FIR do its job. In this sense, the FIR methodology is quite robust and easy to apply.

3.9 Conclusions

The FIR methodology has been introduced, and in particular, its relevance to the task of predicting univariate and multivariate time series of single and/or multiple steps has been discussed. Two time series were initially introduced, one that is predictable, and one that is not. It was shown that FIR is capable

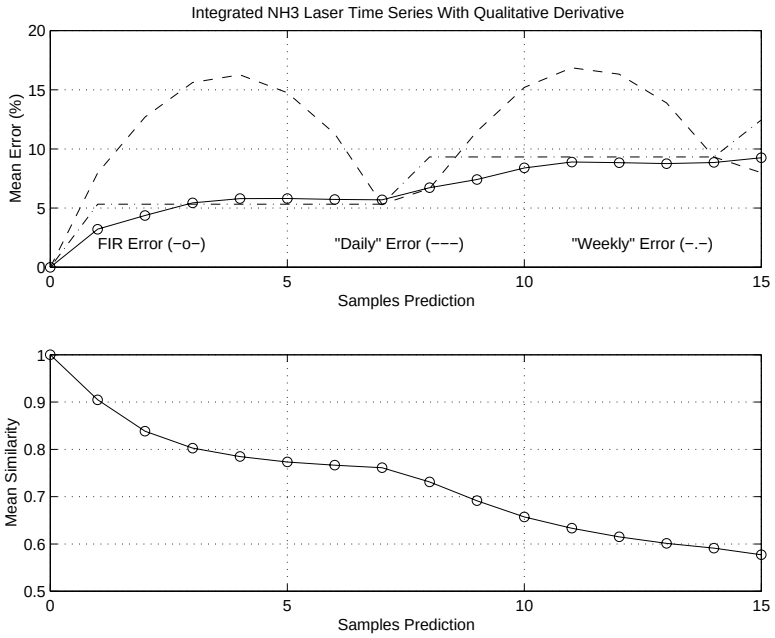

Figure 3.22: Average error and accumulated confidence of Series I using a first-order numerical approximation of the derivative as second input.

of distinguishing between predictable and non–predictable phenomena.

Subsequently, it was shown that FIR filters out what it considers to be noise, a feature that may sometimes be quite useful, but that can also be a nuisance, because the FIR user has little control over what FIR considers noise, and what it considers a signal. Yet, it is important to know about this feature of FIR.

Finally, it was shown that FIR indeed exploits all the information that is given to it in a fairly optimal and robust fashion. To this end, several models were proposed that should have led to improved forecasts if FIR would not already have exploited all the information available to it. Neither of these approaches proved to be useful.

This shows that FIR is indeed a robust modeling methodology that exploits the information it is provided with confidently and reliably without need for much user intervention.

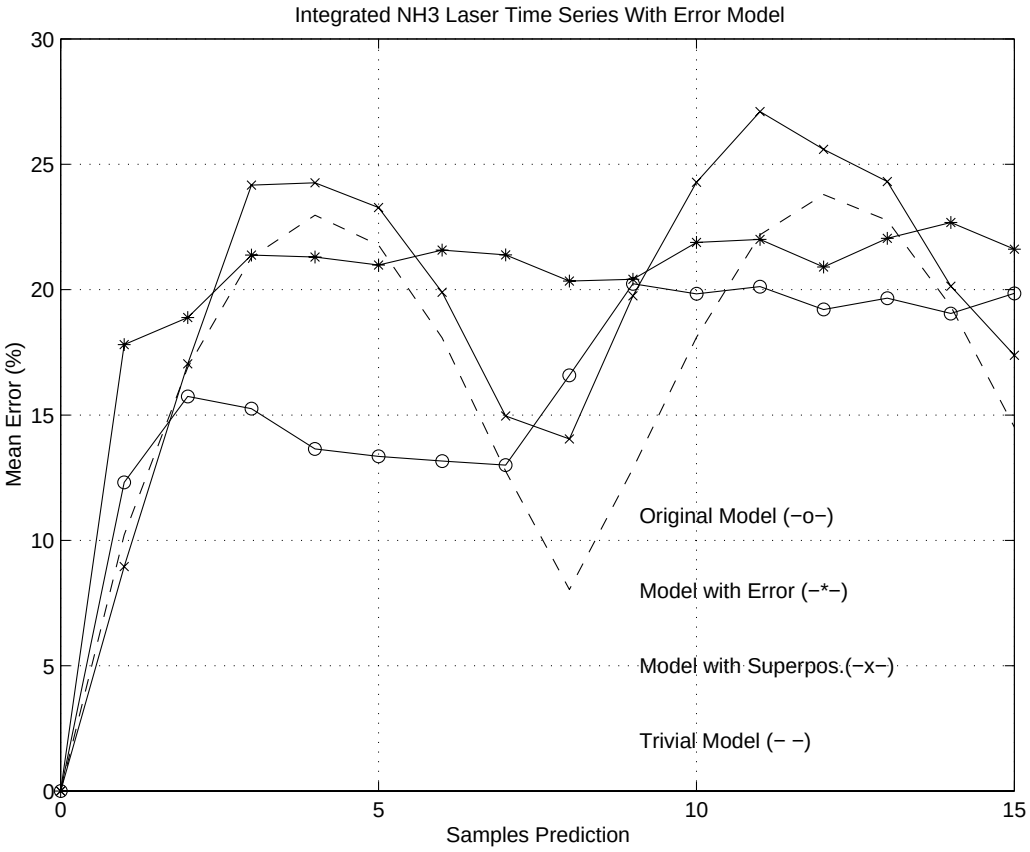

Figure 3.23: Comparison of multi-step prediction errors of models that use a model of the error to gather additional information.

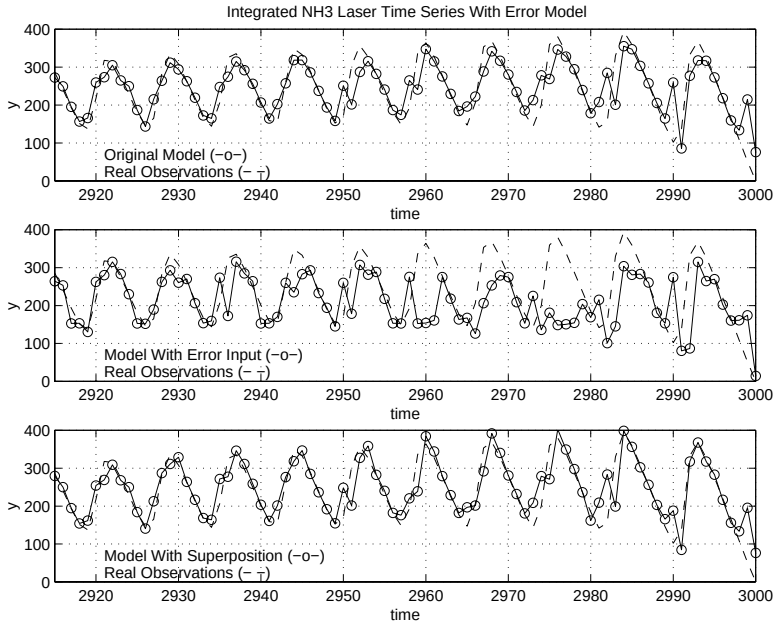

Figure 3.24: Comparison of single-step predictions of different models using models of the error to gather additional information.

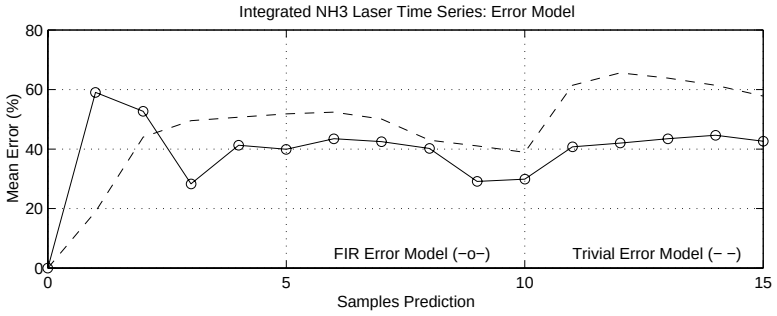

Figure 3.25: Average errors over multi-step predictions of the model that predict the error.

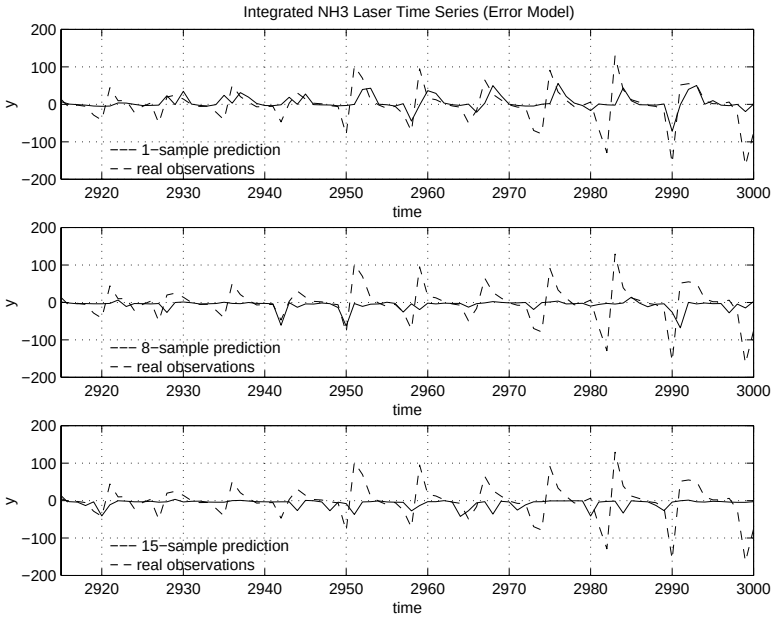

Figure 3.26: Simulation results of single-step and multi-step predictions of the error prediction.

Chapter 4

Comparison of Selected Techniques for Time Series Prediction

4.1 Introduction

Already in Chapter 1, it was mentioned that one of the primary goals of this dissertation is to gain an improved understanding of the virtues as well as shortcomings of the FIR methodology by applying FIR to a problem domain, to which many other methodologies have been applied previously, namely the prediction of univariate time series, so that FIR may be compared with the best among its competitors.

In Chapter 3, FIR was compared to several types of trivial predictors (sometimes also called "naïve predictors"). However, it is to be expected that there exist other techniques that predict much better than these trivial predictors, and FIR needs to be compared with those techniques as well. That is the aim of this chapter.

At first, some of the more prominent of the competitors will be briefly introduced. This discussion is then followed by comparing various of those techniques with FIR in the prediction of two time series that exhibit quite different behavioral patterns: *Series B*, relating to the water demand in the City of Barcelona (Quevedo *et al.* 1988; Griñó 1992; Baggelaar 1992), and *Series R*, describing the water demand in a region of Rotterdam, called the *Berenplaat* (Baggelaar 1992).

4.2 Classification of Prediction Methods

As was outlined already in Chapter 3, all techniques for forecasting the behavior of univariate time series somehow make use of their past behavior, usually in the form of sampled data values dating back over a limited period of time. Mathematically, the forecast can be expressed as:

$$y_t = \tilde{f}(y_{t-1}, y_{t-2}, \cdots, y_{t-d}) \tag{4.1}$$

where d is referred to as the *embedding dimension* of the time series. They differ in how they approximate the unknown function $\tilde{f}$.

Most techniques are *parametric* in nature, i.e., they store the knowledge about the past behavior of the time series in a set of parameter values. FIR is an exception to the rule. FIR is a *non–parametric* method, because it refers, during the forecast, directly to past behavioral patterns that have been observed during the training period.

Among the parametric methods, many are *linear* predictors. In a linear predictor, a linearity assumption is imposed on the unknown function $\tilde{f}$:

$$y_t = \sum_{i=1}^{d} \alpha_i \cdot y_{t-i} \tag{4.2}$$

where α_i are the parameters of the method that themselves can be estimated in many different ways. Such methods are often classified as *auto–regressive (AR) methods* or *infinite impulse response (IIR) filters* (Ljung 1987; Oppenheim and Schafer 1989; Weigend and Gershenfeld 1994).

If the system from which the time series was observed is non–linear, linear predictors may not do a good job at characterizing the behavior of these systems. The reason is that even low–dimensional non–linear systems may exhibit a broad–band power spectrum (Brockwell and Davis 1996). In order to deal better with such systems, non–linear predictors were introduced, such as the *non–linear auto–regressive (NAR) methods* that are based on Volterra series approximations. Most NAR methods limit their non–linearities to bi–linear and quadratic terms:

$$x_t = \sum_{i=1}^{d} \alpha_i \cdot y_{t-i} + \sum_{j=1}^{d} \sum_{k=1}^{j} \alpha_{jk} \cdot x_{t-j} \cdot x_{t-k} \tag{4.3}$$

NAR models are attractive, because, although they are non–linear predictors, they are still *linear in the parameters (LIP)*. The parameters of methods that are of the LIP type can still be estimated using regression techniques.

Other parametric methods are completely non–linear, i.e., they are even non–linear in their parameters. Among those techniques, the most prominent

are the *connectionist methods* that are also referred to as *artificial neural networks (ANN)*. The weights (parameters) of neural networks need to be learned using an iterative learning process, such as *backpropagation training* (Narendra and Parthasarathy 1990; Muller *et al.* 1994; Narendra and Li 1995; Narendra and Mukhopadhyay 1995).

In this chapter, AR, NAR, and ANN models shall be compared to FIR in their capabilities of predicting univariate time series.

4.3 AR Methods

4.3.1 Least Square Estimation

Given a set of n training records, where $n \geq 2 \cdot d$, it is possible to write Eq.(4.2) in a matrix–vector form as follows:

$$\begin{pmatrix} y_{d+1} \\ y_{d+2} \\ \cdots \\ y_n \end{pmatrix} = \begin{pmatrix} y_d & y_{d-1} & \cdots & y_1 \\ y_{d+1} & y_d & \cdots & y_2 \\ \cdots & \cdots & \cdots & \cdots \\ y_{n-1} & y_{n-2} & \cdots & y_{n-d} \end{pmatrix} \cdot \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \cdots \\ \alpha_d \end{pmatrix} \tag{4.4}$$

If $n = 2 \cdot d$, Eq.(4.4) represents a set of d linear equations in d unknowns that can be solved in a unique fashion using any technique suitable for solving linear systems of equations. If $n > 2 \cdot d$, Eq.(4.4) represents an overdetermined set of linear equations in d unknowns that can be solved approximately in a least square sense. Eq.(4.4) can be rewritten as:

$$\mathbf{y} = \mathbf{M} \cdot \mathbf{x} \tag{4.5}$$

where $\mathbf{y} \in R^{(n-d)}$, $\mathbf{M} \in R^{(n-d)\times d}$, and $\mathbf{x} \in R^d$. Therefore:

$$\mathbf{M}' \cdot \mathbf{y} = \mathbf{M}' \cdot \mathbf{M} \cdot \mathbf{x} \tag{4.6}$$

where $\mathbf{M}' \cdot \mathbf{M} \in R^{d\times d}$ is a square matrix that is usually of rank d. Thus:

$$\mathbf{x} = (\mathbf{M}' \cdot \mathbf{M})^{-1} \cdot \mathbf{M}' \cdot \mathbf{y} \tag{4.7}$$

is an approximate solution of the set of equations, where $(\mathbf{M}' \cdot \mathbf{M})^{-1} \cdot \mathbf{M}'$ is a pseudo–inverse of $\mathbf{M}$. In Matlab, this solution can be obtained using the backslash operator:

$$\mathbf{x} = \mathbf{M} \backslash \mathbf{y} \tag{4.8}$$

a notation that, for reasons of convenience, has meanwhile been adopted in the linear algebra literature throughout.

Once the coefficient vector $\mathbf{x}$ has been found, future predictions can recursively be obtained using the equation:

$$y_{n+k} = \mathbf{Y} \cdot \mathbf{x} \tag{4.9}$$

where:

$$\mathbf{Y} = \begin{pmatrix} y_{n+k-1} & y_{n+k-2} & \dots & y_{n+k-d} \end{pmatrix} \tag{4.10}$$

This concludes the straightforward description of the method.

It is of interest to discuss the *stability* of an AR model. To this end, it is useful to represent Eq.(4.2) in the frequency domain. Before doing so, it is advisable to represent the *approximation error* as an additive term on the right–hand side:

$$y(t) = \sum_{i=1}^{d} \alpha_i \cdot y_{t-i} + e(t) \tag{4.11}$$

where $y(t)$ is the true value of y at time t, whereas y_t (of Eq.(4.2)) is the approximation of $y(t)$ at time t, thus:

$$y(t) - y_t = e(t) \tag{4.12}$$

is the error, $e(t)$, committed by the approximation, y_t, at time t.

Eq.(4.11) can now be transformed into the frequency domain:

$$Y(z) = \sum_{i=1}^{d} \alpha_i \cdot z^{-i} \cdot Y(z) + E(z) \tag{4.13}$$

where $z = e^{Ts}$ is the z–operator of the classical z–transform. Thus:

$$\left(1 - \sum_{i=1}^{d} \alpha_i \cdot z^{-i}\right) \cdot Y(z) = E(z) \tag{4.14}$$

or:

$$Y(z) = \frac{z^n}{z^n - \sum_{i=1}^{d} \alpha_i \cdot z^{n-i}} \cdot E(z) \tag{4.15}$$

Hence $y(t)$ is stable iff all the poles of the denominator polynomial: $z^n - \sum_{i=1}^{d} \alpha_i \cdot z^{n-i}$ are inside the unit circle of the complex z–plane (Ogata 1970; Kuo 1991).

There is no guarantee that the least squares approach to determining the parameter values of the AR model will satisfy the stability requirement, i.e., recursive predictions using Eq.(4.9) may grow beyond all bounds.

4.3.2 Autocorrelation

If the univariate time series is *stationary*, it seems reasonable to expect that recursive predictions produce a stationary forecast as well. This means that the model of Eq.(4.15) should be marginally stable, i.e., the dominant pole should be at $z = 1.0$. One way to ensure that there is a pole at $z = 1.0$ is to request that:

$$\sum_{i=1}^{d} \alpha_i = 1.0 \tag{4.16}$$

because, in this case, the denominator of Eq.(4.15) will become zero for $z = 1.0$.

Eq.(4.16) makes practical sense. It means that the forecast y_t is a linear blend (a moving average) of the previous d values. Since the expectation value of any data point y_{t-i} is constant, as the series is assumed to be stationary, the expectation value of any moving average is also constant and equal to that of y_{t-i}, i.e.:

$$\mathrm{E}\{y_t\} = \mathrm{E}\{y_{t-i}\} \quad \forall i \tag{4.17}$$

The forecast is stationary if all other poles are inside the unit circle, i.e., if the model is indeed marginally stable.

One way to determine a decent set of parameter values is to make use of the autocorrelation function $\rho(t)$ of the time series $y(t)$. The value ρ_i represents the relative importance of y_{t-i} in the approximation of y_t. Thus, the autocorrelation function can be used to determine the embedding dimension, d, by ignoring all values of i for which ρ_i has decayed to a value below significance. The coefficients can then be estimated using the relative importance of the remaining ρ_i factors. Let:

$$s_\rho = \sum_{i=1}^{d} \rho_i \tag{4.18}$$

Then:

$$\alpha_i = \frac{\rho_i}{s_\rho} \tag{4.19}$$

provides a meaningful set of coefficients satisfying Eq.(4.16). It can be checked easily whether the coefficients obtained in this fashion satisfy the stability condition. Sometimes, this method is referred to as *Yule–Walker* method, because the autocorrelation coefficients are related to the prediction model by means of the Yule–Walker equations (Yule 1927).

4.3.3 FIR Weights

Both the least squares method and the autocorrelation method have the disadvantage that they provide linear parameter estimators. If the system from which the time series was observed is non–linear, a linear estimator may produce results that are far from optimal.

Another technique that may exploit non–linear characteristics better is to use FIR to estimate the parameter values of the AR model. To this end, the following set of masks may be proposed:

$$m_1 = \begin{pmatrix} -1 \\ +1 \end{pmatrix} \;;\; m_2 = \begin{pmatrix} -1 \\ 0 \\ +1 \end{pmatrix} \;;\; m_3 = \begin{pmatrix} -1 \\ 0 \\ 0 \\ +1 \end{pmatrix} \;;\; \ldots \;;\; m_d = \begin{pmatrix} -1 \\ 0 \\ \ldots \\ 0 \\ +1 \end{pmatrix} \tag{4.20}$$

The quality of each of these masks of complexity 2 can be evaluated. These mask qualities shall be referred to as $Q_1 \ldots Q_d$. As in the case of the autocorrelation coefficients, the quality Q_i is a measure of the relative importance of y_{t-i} in approximating y_t. Thus, it makes sense to compute:

$$s_{FIR} = \sum_{i=1}^{d} Q_i \tag{4.21}$$

and then:

$$\alpha_i = \frac{Q_i}{s_{FIR}} \tag{4.22}$$

which constitute another set of meaningful α_i coefficients that satisfy Eq.(4.16). As before, the stability condition needs to be verified separately.

4.3.4 Neural Networks

Yet other approaches make use of ANNs to estimate the parameters of an AR model. The idea is simple. Starting out from an initial set of parameters,

obtained using any of the aforementioned methods, it is possible to check how well the so found parameters approximate the training data set or any other data set that has not been used yet. Then, the parameters can be modified iteratively in order to optimize the approximation using any given performance index.

One way to solve this optimization problem is to create a neural network that exhibits the parameters of the AR model as its outputs, and train the neural network in a supervised training mode in order to optimize the desired performance index.

This thesis shall not deal with such hybrid methods. Details can be found in the open literature (Weigend and Gershenfeld 1994; Zimmermann and Weigend 1995; Delgado 1998).

4.4 ARMA Methods

Usually, the number of training records of an AR model is much larger than $2 \cdot d$. Therefore, Eq.(4.8) can only be solved in an approximate fashion. Using:

$$\mathbf{x} = \mathbf{M} \backslash \mathbf{y(t)} \tag{4.23}$$

where $\mathbf{y(t)}$ is the vector of true measurement values, then inverting Eq.(4.23):

$$\mathbf{y_t} = \mathbf{M} \cdot \mathbf{x} \tag{4.24}$$

one obtains only an approximation of $\mathbf{y(t)}$, denoted as $\mathbf{y_t}$. The error of this approximation is given by Eq.(4.12) rewritten as:

$$\mathbf{e(t)} = \mathbf{y(t)} - \mathbf{y_t} \tag{4.25}$$

where $\mathbf{e(t)}$ is a vector of sampled values of the true error $e(t)$. Thus, Eq.(4.25) can also be written as an equation between signals, rather than between samples:

$$e(t) = y(t) - y_t \tag{4.26}$$

The signal $e(t)$ can also be interpreted as a univariate time series. It is the time series of the error between the true time series and its approximation using the AR model.

The same approach that was used to model $y(t)$ resulting in its approximation y_t can also be applied to the time series of the error, $e(t)$:

$$e_t = \sum_{k=1}^{d_e} \beta_k \cdot e_{t-k} \tag{4.27}$$

leading to an approximation e_t of the true error $e(t)$. The embedding dimension of this model is d_e, and the coefficients shall be called β_k. d_e can be different from d.

Any of the aforementioned methods can be used to estimate the parameters β_k. Notice that the problem of identifying a model of the error is exactly the same as that of identifying the model of the original time series, yet it has become customary to refer to this type of model as *moving average (MA) model* or *finite impulse response (FIR) filter*, rather than *autoregressive (AR)* model or *infinite impulse response (IIR) filter*.

Methodologically, this distinction does not seem justified, and the name "moving average" is confusing, because both models are in fact moving average models, but tradition overrules whatever objections there may be.

The two models (RA and MA) can now be combined. Using Eq.(4.11) once more:

$$y(t) = \sum_{i=1}^{d} \alpha_i \cdot y_{t-i} + e(t) \tag{4.28}$$

it is possible to obtain an improved prediction of the original time series by replacing the true error by its approximation:

$$\hat{y}_t = \sum_{i=1}^{d} \alpha_i \cdot y_{t-i} + e_t \tag{4.29}$$

or:

$$\hat{y}_t = \sum_{i=1}^{d} \alpha_i \cdot y_{t-i} + \sum_{k=1}^{d_e} \beta_k \cdot e_{t-k} \tag{4.30}$$

Hopefully, the error of the improved prediction, $\hat{y}_t$, is smaller than that of the original prediction, y_t, thus:

$$\|y(t) - \hat{y}_t\| < \|y(t) - y_t\| \tag{4.31}$$

The improved model is called ARMA model.

4.5 ARIMA Methods

One problem that has not been addressed yet is the occurrence of periodic or quasi–periodic behavior as it can be found in some time series. There is nothing in either AR or ARMA models that would guarantee the preservation of periodic or quasi–periodic behavioral patterns in the predictions. ARIMA models overcome this deficiency. Notice that ARIMA models are simply a

special case of ARMA models that can furthermore only be used in case of periodic or quasi–periodic time series.

The idea behind the ARIMA models is again quite simple. Assuming that the time series exhibits a period (or quasi–period) of τ, where τ is a multiple of the sampling rate. It is then possible to construct a new time series:

$$z(t) = y(t) - y(t - \tau) \tag{4.32}$$

that no longer contains the periodic behavior. In fact, $z(t)$ is an error model that only reflects the deviations from the periodic behavior, because, if the behavior of the time series had been truly periodic in τ, $z(t)$ would be 0.0 throughout.

It is now possible to generate an ARMA model of the time series $z(t)$ using any of the techniques proposed earlier. The approximation z_t may denote the ARMA prediction of $z(t)$. It makes sense to write:

$$z_t = y_t - y(t - \tau) \tag{4.33}$$

For simplicity, τ is now interpreted as an index of displacement rather than a true time value. Thus, Eq.(4.32) can be transformed into the frequency domain as follows:

$$Z(z) = Y(z) - z^{-\tau} \cdot Y(z) = \left(1 - z^{-\tau}\right) \cdot Y(z) \tag{4.34}$$

or:

$$Y(z) = \frac{z^{\tau}}{z^{\tau} - 1} \cdot Z(z) \tag{4.35}$$

Combining Eq.(4.35) with Eq.(4.15), the following ARI model of the original time series is obtained:

$$Y(z) = \frac{z^{n} \cdot z^{\tau}}{(z^{n} - \sum_{i=1}^{d} \alpha_i \cdot z^{n-i}) \cdot (z^{\tau} - 1)} \cdot E(z) \tag{4.36}$$

The stability properties of the new model are not modified in an essential fashion by the additional term $z^{\tau} - 1$ in the denominator. The new term only adds another τ marginally stable poles placed at equal angles around the unit circle of the complex z–domain. However, it now makes sense to request that all poles of the AR model of $z(t)$ are clearly *inside* the unit circle to avoid having to deal with a double pole at $z = 1.0$, as this can cause drift. The MA portion of the ARIMA model is harmless, as it only adds another numerator polynomial to Eq.(4.36).

The same general approach can also be used to deal with linear non–stationary behavior. For example, if $y(t)$ has a positive constant trend, then:

$$z(t) = y(t) - y(t-1) \tag{4.37}$$

will have no trend any longer, i.e., will be stationary. If the trend is non–linear, e.g. describing exponential growth phenomena, stationarity can be achieved using non–linear transforms. For example, exponential growth can be eliminated using the transformation:

$$z(t) = \frac{y(t) - y(t-1)}{y(t)} \tag{4.38}$$

as shown in (Moorthy *et al.* 1998).

4.6 NAR and NARMA Methods

As NAR models are of the LIP type, least squares can still be used to identify the parameters of Eq.(4.3):

$$x_t = \sum_{i=1}^{d} \alpha_i \cdot y_{t-i} + \sum_{j=1}^{d} \sum_{k=1}^{j} \alpha_{jk} \cdot x_{t-j} \cdot x_{t-k} \tag{4.39}$$

The equations can be written in matrix–vector form as follows:

$$\begin{pmatrix} y_{d+1} \\ y_{d+2} \\ \cdots \\ y_n \end{pmatrix} = \begin{pmatrix} y_d & \cdots & y_1 & y_d \cdot y_d & \cdots & y_1 \cdot y_1 \\ y_{d+1} & \cdots & y_2 & y_{d+1} \cdot y_{d+1} & \cdots & y_2 \cdot y_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ y_{n-1} & \cdots & y_{n-d} & y_{n-1} \cdot y_{n-1} & \cdots & y_{n-d} \cdot y_{n-d} \end{pmatrix} \cdot \begin{pmatrix} \alpha_1 \\ \cdots \\ \alpha_d \\ \alpha_{11} \\ \cdots \\ \alpha_{dd} \end{pmatrix} \tag{4.40}$$

where $n \geq \frac{d \cdot (d+1)}{2} + 2 \cdot d$. Although the same least squares method can be used as in the AR case to identify the $\frac{d \cdot (d+1)}{2} + d$ parameters of the NAR model, usually another approach is chosen. The reason is that most researchers prefer to operate on a model with considerably fewer parameters, as a highly parameterized model has the tendency to identify the noise. Such a model leads to excellent results when applied to the training data, but to poor results when applied to a hitherto unseen set of data.

Several suboptimal search techniques can be employed to identify a meaningful subset of NAR parameters.

The *aggregation method* starts out with a full set of parameters identified from Eq.(4.40). The parameters are identified as follows:

$$\mathbf{y} = \mathbf{M} \cdot \mathbf{x} \tag{4.41}$$

where $\mathbf{y} \in R^{(n-d)}$, $\mathbf{M} \in R^{(n-d)\times(\frac{d\cdot(d+1)}{2}+d)}$, and $\mathbf{x} \in R^{(\frac{d\cdot(d+1)}{2}+d)}$. Therefore:

$$\mathbf{x} = \mathbf{M} \backslash \mathbf{y}(\mathbf{t}) \tag{4.42}$$

is the set of parameters. The error of the prediction can be computed as follows:

$$\mathbf{e} = \mathbf{y} - \mathbf{M} \cdot \mathbf{x} \tag{4.43}$$

with the norm $\|\mathbf{e}\|$. The process can now be repeated, each time leaving out one of the parameters (one column of $\mathbf{M}$ and the corresponding row of $\mathbf{x}$). The corresponding error norm $\|\mathbf{e_i}\|$ or $\|\mathbf{e_{jk}}\|$, where the index (i or jk) indicates the omitted parameter, is expected to be larger than $\|\mathbf{e}\|$. However, the increase in the error should not be large, because the model still contains many parameters.

The aggregation method eliminates the parameter that *least increases* the error permanently, and then continues eliminating one of the remaining parameters. The process is repeated until the error norm starts growing rapidly. Plotting the error norm *vs.* the number of omitted parameters, the resulting curve usually shows a knee. The optimal set of parameters, k, is just below the knee of the curve. Figure 4.1 shows a sketch of a typical error norm function plotted *vs.* the number of parameters omitted.

The *refinement method* starts out with a single parameter. It tries out one parameter at a time, computing the error norms $\|\mathbf{e_i}\|$ or $\|\mathbf{e_{jk}}\|$. Here, the index (i or jk) indicates the added parameter.

The refinement method then keeps the parameter that *most decreases* the error permanently, and then continues adding another parameter. The process is repeated until the error norm stops decreasing rapidly. Plotting the error norm *vs.* the number of added parameters, the resulting curve usually shows a knee. The optimal set of parameters, j, is just beyond the knee of the curve. Figure 4.2 shows a sketch of a typical error norm function plotted *vs.* the number of parameters added.

Once the NAR portion of the model has been found, the error can be computed, and an AR model of the error (the MA portion) can be added in just the same way as for the ARMA and ARIMA models. Although it would be possible to construct also a NAR model of the error, this is hardly ever done.

Obviously, the neat stability analysis of linear predictors does not apply to NAR and/or NARMA models. Thus, it is much more difficult to prove

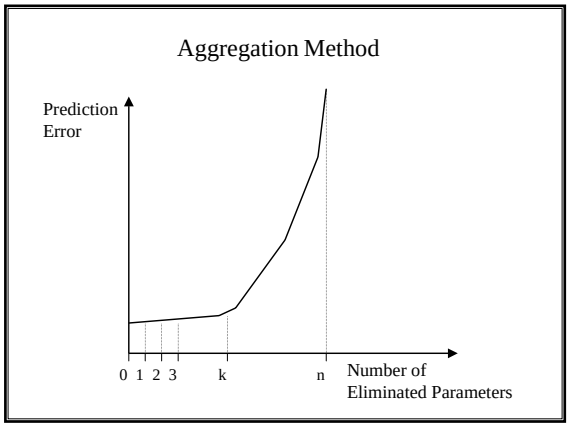

Figure 4.1: Aggregation method.

that a NAR (or NARMA) forecast of a stationary time series is stationary than was the case for AR (or ARMA) models.

4.7 ANN Methods

Even NARMA models are limited in the types of non–linearities that they support. Neural networks offer a way out of this limitation. They start out with Eq.(4.1):

$$y_t = \tilde{f}(y_{t-1}, y_{t-2}, \cdots, y_{t-d}) \tag{4.44}$$

and try to fit an arbitrarily non–linear model, i.e., the true (unknown) function $\tilde{f}$ to the training data. Since also neural networks are parameterized, *some* structural assumption must be made. However, the assumption made is so general that it fits *any* function $\tilde{f}$. There are proofs in the open literature that show that a feedforward neural network with at least one hidden layer can fit any arbitrarily non–linear univalued function (Haber and Unbehauen 1990; Connor *et al.* 1992; Cottrell *et al.* 1995; Golob *et al.* 1998).

How feedforward neural networks are constructed is the topic of many books and articles. It is beyond the scope of this dissertation to even attempt to review this literature. However, since neural networks have been rather successful in predicting time series (Ghoshray 1996; Delgado

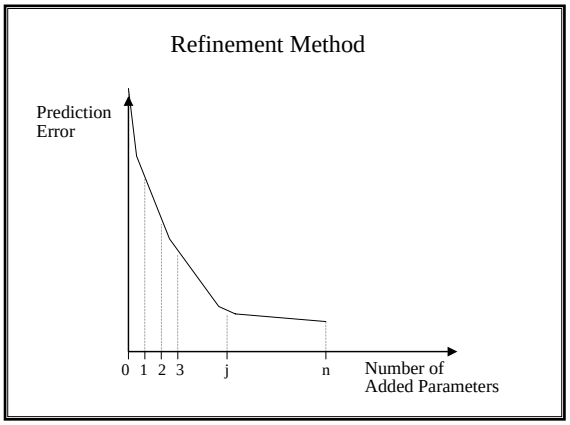

Figure 4.2: Refinement method.

1998), this dissertation would be incomplete without at least a comparison of the success of FIR *vs.* ANN when used to predict a few time series of different characteristics. To this end, an ANN was constructed using Matlab, a software developed at (MathWorks 1997).

4.8 Time Series B: Barcelona Water Demand

Time Series B represents the water demand of an area of the city of Barcelona (Aigües de Barcelona 1985). The measurement data are shown in Figure 4.3.

The characteristics of this time series are presented in Table 4.1. Series B is mildly non–stationary. During the observed period, the water consumption grew slightly, either because the city is still growing, or because the average household consumes more water (e.g. due to a wider proliferation of dish washers), or finally, because newer production facilities require slightly more water. Although FIRs performance is not affected by the mildly non–stationary nature of this time series, some of the contending methods are. Series B is definitely time varying. On Sundays and public holidays, Barcelona consumes considerably less water than on regular work days. Moreover, the month of August is vacation month in Barcelona, which is reflected in a significant reduction in the water consumption patterns during that month. Series B is mildly stochastic, yet the data can be considered

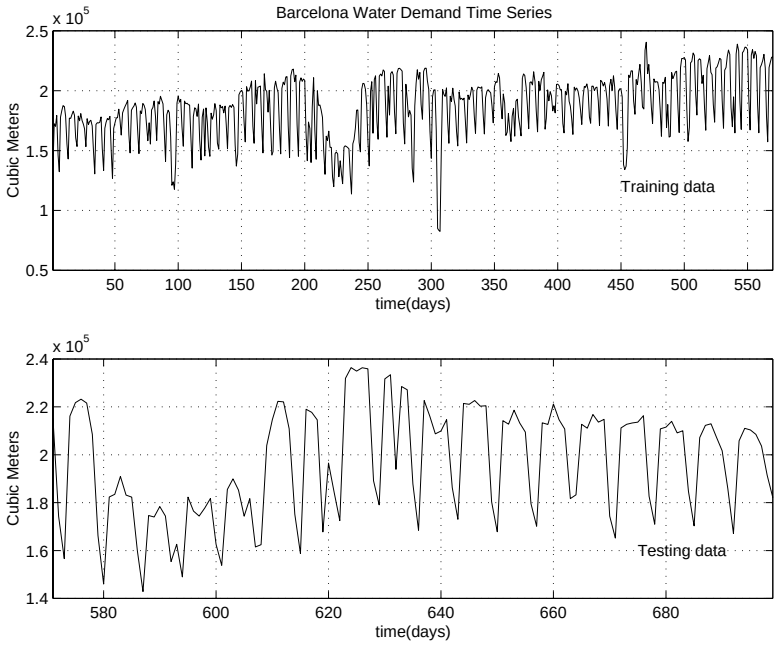

Figure 4.3: Barcelona water demand: Training and testing data.

fairly clean. Only 700 data points are available, i.e., the time series is relatively short. 1.5 years worth of daily measurements, from January 1985 to July 1986, were available to generate the model.

The auto–correlation of this time series is shown in Figure 4.4. Even by naked eye, it is quite easy to discern a strong weekly cycle.

4.8.1 FIR Qualitative Simulation

Due to the cyclic nature of Series B, it was decided to choose a mask depth of two weeks in constructing the FIR model. The optimal FIR model for this time series was found to be:

Table 4.1: Classification of Time Series B.

natural	B	synthetic	
stationary		non–stationary	B
time invariant		time varying	B
low dimensional		stochastic	B
clean	B	noisy	
short	B	long	
dormant		active	B
documented	B	blind	
linear		non–linear	B
scalar	B	vector	
single recording	B	multiple recordings	
continuous	B	discrete	

$$\begin{array}{l} \\ t-14\delta t \\ t-13\delta t \\ \cdots \\ t-8\delta t \\ t-7\delta t \\ t-6\delta t \\ \cdots \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \left(\begin{array}{r} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ 0 \\ -3 \\ +1 \end{array}\right) \end{array} \tag{4.45}$$

This result is quite reasonable. Due to the strong weekly cycle inherent in this time series, FIR concludes that the most useful data points to predict today's water demand are yesterday's water demand, last week's water demand, and the water demand two weeks ago.

570 days (from January 1, 1985 to July 24, 1986) were used as training data, whereas 128 days (from July 25, 1986 to November 29, 1986) were used as testing data. Thanks to the strong auto–correlation of this time series, 570 data points were sufficient to derive a model exhibiting fairly good short–term prediction capabilities.

A prediction matrix (cf. Matrix (3.29)) with 16 columns was constructed, i.e., at each time instant, a multi–step prediction over 15 days was performed. The average error and the average accumulated confidence are plotted in

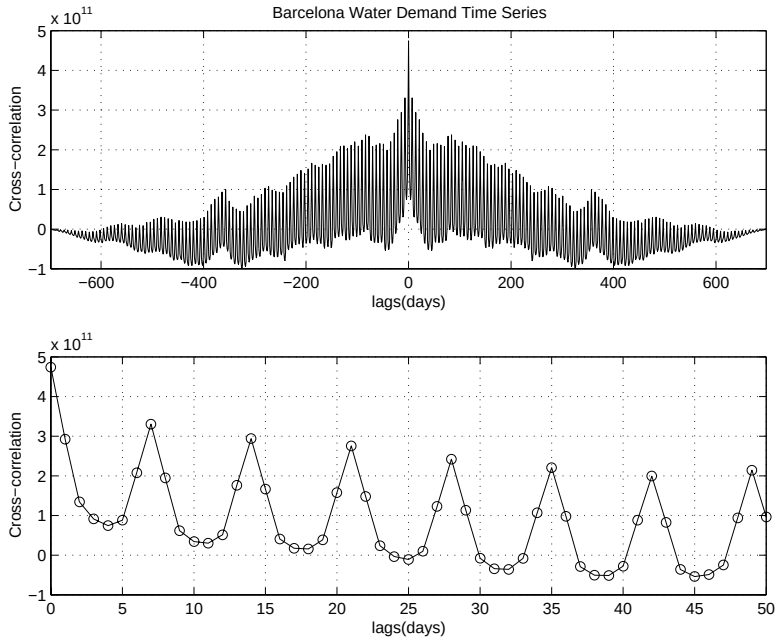

Figure 4.4: Auto–correlation of Barcelona water demand data.

Figure 4.5[1].

As a gauge, the error is compared to those of the *trivial daily prediction* and *trivial weekly prediction* as introduced in Chapter 3 of this dissertation.

The results are somewhat sobering. Only for the one–step prediction, which is the most useful in this application as the water company wants to plan always one day ahead, FIR predicts significantly better than the weekly trivial predictor. Already FIR's two–step prediction is about equal in quality to the weekly trivial prediction. Some of the predictions are even slightly worse, because the FIR prediction, contrary to the trivial predictions, does not fully preserve the statistical parameters of the series. The FIR prediction reduces the standard deviation as it filters out what it considers to be noise.

Figure 4.6 compares the one–day prediction, the eight–day prediction, and the fifteen–day prediction with the measurement data.

The reduction in forecast quality is quite noticeable, yet even a two–week forecast is still somewhat meaningful. It is better than both of the trivial predictors for the same forecasting period.

[1]The results shown in Figure 4.5 were obtained with an improved FIR algorithm, the details of which will only be revealed in Chapter 6 of this dissertation.

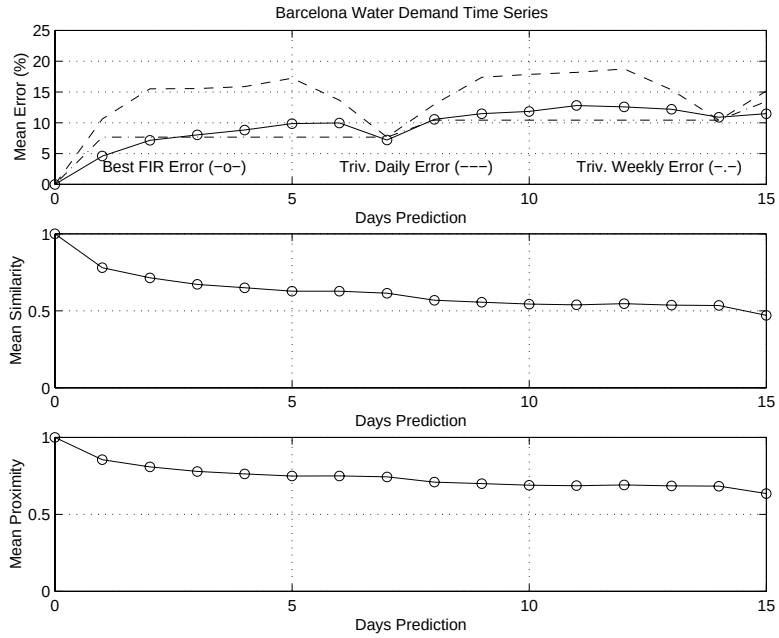

Figure 4.5: Barcelona water demand multiple–step simulation using FIR.

4.8.2 AR Predictions

In the sequel, FIR is being compared against three different AR models, one using least squares, the second using the autocorrelation coefficients, and the third using the FIR mask qualities. Although the embedding dimension of the FIR model had been chosen to be $d = 14$, it was decided to base the AR models on a single week only, i.e., the embedding dimension was reduced to $d = 7$.

Least Squares Method

Two separate least square AR models were constructed. The first model was based on the equation:

$$y_t = \sum_{i=1}^{7} \alpha_i \cdot y_{t-i} \tag{4.46}$$

The α_i coefficients were identified using the entire training data set. The resulting model was:

$$y_t = 0.5278 \cdot y_{t-1} - 0.1467 \cdot y_{t-2} + 0.1130 \cdot y_{t-3}$$

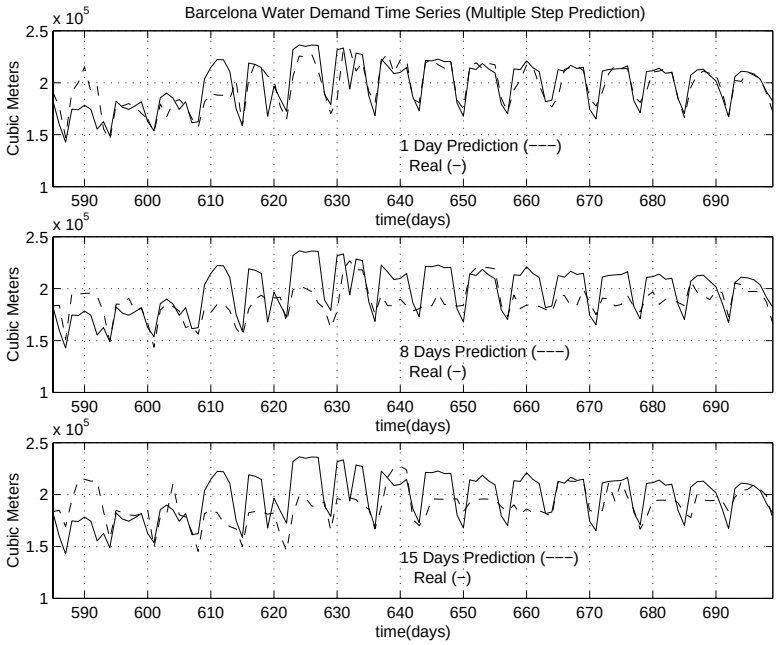

Figure 4.6: Barcelona water demand multiple–step simulation using FIR.

$$\begin{aligned} &-0.0237 \cdot y_{t-4} - 0.0661 \cdot y_{t-5} + 0.1658 \cdot y_{t-6} \\ &+0.4308 \cdot y_{t-7} \end{aligned} \tag{4.47}$$

As was to be expected, the coefficients associated with the previous day and seven days before are most prominent. This model was then used in a *simulation mode* to predict the future behavior of the time series over 15 days.

The sum of the parameters is $s_\alpha = 1.0009$, i.e., although the value is close to 1.0, it is to be expected that there is at least one pole slightly outside the unit circle. The poles of the denominator polynomial of Eq.(4.15) are:

$$r = \begin{pmatrix} 1.0002 \\ 0.6293 + 0.7143 \cdot \mathrm{i} \\ 0.6293 - 0.7143 \cdot \mathrm{i} \\ -0.1809 + 0.8801 \cdot \mathrm{i} \\ -0.1809 - 0.8801 \cdot \mathrm{i} \\ -0.6845 + 0.3467 \cdot \mathrm{i} \\ -0.6845 - 0.3467 \cdot \mathrm{i} \end{pmatrix} \tag{4.48}$$

with the absolute values:

$$r_{\text{abs}} = \begin{pmatrix} 1.0002 \\ 0.9519 \\ 0.9519 \\ 0.8985 \\ 0.8985 \\ 0.7673 \\ 0.7673 \end{pmatrix} \tag{4.49}$$

thus indeed, one eigenvalue is slightly outside the unit circle, whereas all other poles are clearly within the unit circle.

Figure 4.7 compares the one–day prediction, the eight–day prediction, and the fifteen–day prediction with the measurement data.

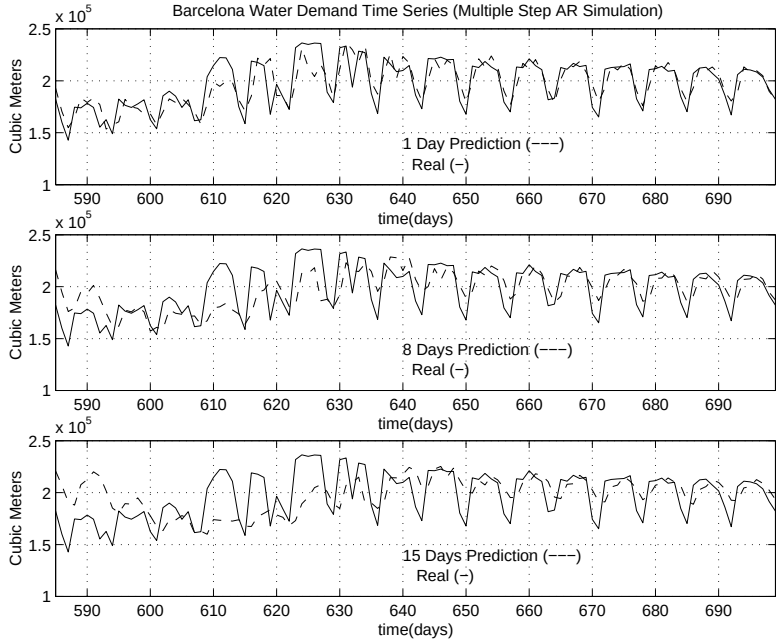

Figure 4.7: Multiple Step Barcelona water demand simulation using least squares.

In spite of the slightly unstable pole, the periodic variations of the 15–day prediction seem to be somewhat smaller than those of the 1–day prediction.

The second approach used fifteen different models. The first model is that of Eq.(4.46). However, this model is only used to predict over a single day. For the second day prediction, a modified model was used:

$$y_{t+1} = \sum_{i=1}^{7} \beta_i \cdot y_{t-i} \tag{4.50}$$

i.e., the water consumption of the second day is predicted directly using measurement data that lie at least two days behind. The coefficients found were:

$$\begin{aligned} y_{t+1} \quad = \quad & 0.2953 \cdot y_{t-1} - 0.0149 \cdot y_{t-2} + 0.0460 \cdot y_{t-3} \\ & -0.0345 \cdot y_{t-4} + 0.0549 \cdot y_{t-5} + 0.7737 \cdot y_{t-6} \\ & -0.1204 \cdot y_{t-7} \end{aligned} \tag{4.51}$$

As was to be expected, the most important days are now two days back, seven days back, and eight days back. Just by chance, the sum of the parameters here is $s_\beta = 1.0000$, with the poles located at:

$$r = \begin{pmatrix} 1.0000 \\ 0.5002 + 0.8300 \cdot \mathrm{i} \\ 0.5002 - 0.8300 \cdot \mathrm{i} \\ -0.4717 + 0.8284 \cdot \mathrm{i} \\ -0.4717 - 0.8284 \cdot \mathrm{i} \\ -0.9158 \\ 0.1541 \end{pmatrix} \tag{4.52}$$

with the absolute values:

$$r_{\mathrm{abs}} = \begin{pmatrix} 1.0000 \\ 0.9691 \\ 0.9691 \\ 0.9532 \\ 0.9532 \\ 0.9158 \\ 0.1541 \end{pmatrix} \tag{4.53}$$

Similarly for the three–day to fifteen–day predictions. In all these models, the sums of the coefficients assume values in the vicinity of 1.0, with the largest pole being in the vicinity of $z = 1.0$, either slightly larger or slightly smaller, and all other poles being clearly inside the unit circle.

Contrary to the previous approach, this forecast is a pure *prediction*, as no previously predicted values are ever being used for future forecasts.

Figure 4.8 compares the one–day prediction, the eight–day prediction, and the fifteen–day prediction with the measurement data.

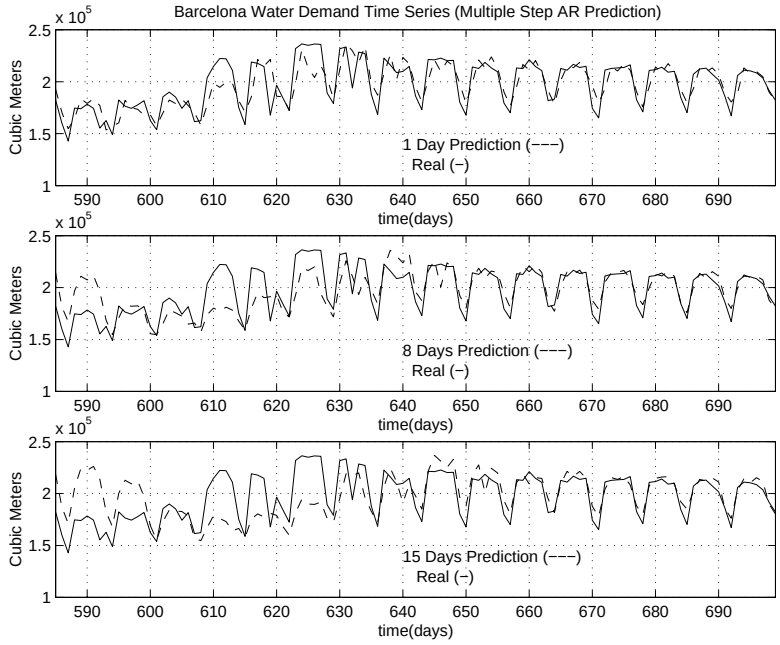

Figure 4.8: Multiple Step Barcelona water demand prediction using least squares.

The one–day prediction is identical to that of the *simulation* approach, but the eight–day and fifteen–day predictions are clearly better.

Figure 4.9 compares the average errors with those of the FIR prediction. The *prediction mode* clearly outperforms the *simulation mode*. The reason is that subsequent contaminations of future forecasts by using previously predicted values cause more damage than the larger horizon of the direct prediction forecasts.

Both techniques are definitely inferior to FIR. They are also inferior to the trivial weekly prediction. The reason is that the AR model compresses the entire knowledge about the training data into seven parameter values. This is only meaningful if the time series is truly *stationary*. Looking at Figure 4.3, it is quite evident that this assumption does not hold at all. The water consumption seems to constantly grow over the 1.5 year training period. Thus, the parameter values contain information that is no longer truly relevant.

The author tried to prefilter the data by subtracting a linear regression line (obtained by looking at the training data only) from the measurement

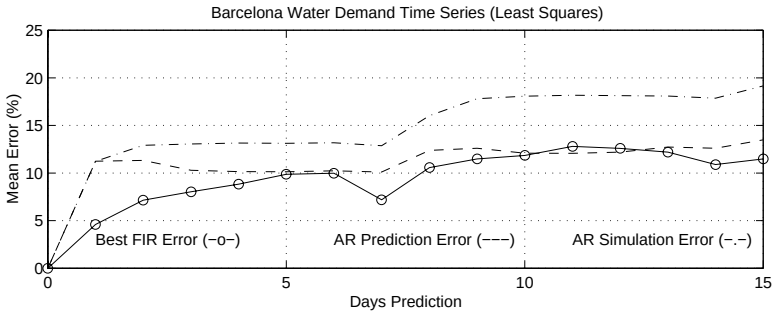

Figure 4.9: Comparative analysis of Barcelona water demand predictions: Least squares *vs.* FIR.

data, and also by subtracting a "standard" week (consisting of averages over each weekday of the training data) from the original time series. The reader will be saved from having to look at the results. They were even worse! The reason is that, during the testing period, for the first time in a long while, the consumption seems to decrease, i.e., neither the regression line nor the average week help with detrending the data. Of course, it would have been possible to use a differentiation approach instead, but this would lead to an ARI model, which shall be discussed later.

FIR did not suffer from the mildly non–stationary characteristics of the time series. Due to the five–nearest neighbor rule, FIR only looks at very similar patterns in making forecasts, and therefore, variations that lie within the range of the training data do not cause insurmountable problems to the FIR methodology.

Autocorrelation Method

The idea behind the autocorrelation method had been explained earlier in this chapter. Also the autocorrelation method can be implemented either in a simulation mode or in a prediction mode.

In the *simulation mode*, the prediction at time t is a linear combination of the values obtained for the previous seven days. The resulting model is:

$$\begin{aligned} y_t \;=\; & 0.3505 \cdot y_{t-1} + 0.2170 \cdot y_{t-2} + 0.1008 \cdot y_{t-3} \\ & +0.0659 \cdot y_{t-4} + 0.0514 \cdot y_{t-5} + 0.0649 \cdot y_{t-6} \\ & +0.1495 \cdot y_{t-7} \end{aligned} \tag{4.54}$$

All coefficients are positive, since the first seven autocorrelation coefficients are all positive.

This time, there is certainly one pole at $z = 1.0$, since the sum of all coefficients has been normalized to 1.0. The pole locations are:

$$r = \begin{pmatrix} 1.0000 \\ 0.4886 + 0.5698 \cdot \mathrm{i} \\ 0.4886 - 0.5698 \cdot \mathrm{i} \\ -0.6480 + 0.3039 \cdot \mathrm{i} \\ -0.6480 - 0.3039 \cdot \mathrm{i} \\ -0.1653 + 0.7004 \cdot \mathrm{i} \\ -0.1653 - 0.7004 \cdot \mathrm{i} \end{pmatrix} \tag{4.55}$$

with the absolute values:

$$r_{\mathrm{abs}} = \begin{pmatrix} 1.0000 \\ 0.7506 \\ 0.7506 \\ 0.7157 \\ 0.7157 \\ 0.7196 \\ 0.7196 \end{pmatrix} \tag{4.56}$$

Hence all poles are inside the unit circle except for the single marginally–stable pole at $z = 1.0$. Multi–step predictions are obtained by reusing previous predictions in the process of making new predictions.

In the *prediction mode*, autocorrelation values from earlier days are being used. For example, the model of the two–day prediction is:

$$\begin{aligned} y_{t+1} \;=\; & 0.2458 \cdot y_{t-1} + 0.1142 \cdot y_{t-2} + 0.0747 \cdot y_{t-3} \\ & +0.0582 \cdot y_{t-4} + 0.0735 \cdot y_{t-5} + 0.1693 \cdot y_{t-6} \\ & +0.2644 \cdot y_{t-7} \end{aligned} \tag{4.57}$$

where the coefficients are the normalized autocorrelation values of two up to eight days away from the center.

Figure 4.10 compares the average errors with those of the FIR prediction. Just as in the least squares case, the *prediction mode* outperforms the *simulation mode* because of data contamination problems. Although the errors remain below 25% at all times, the results are consistently worse than those obtained using least squares.

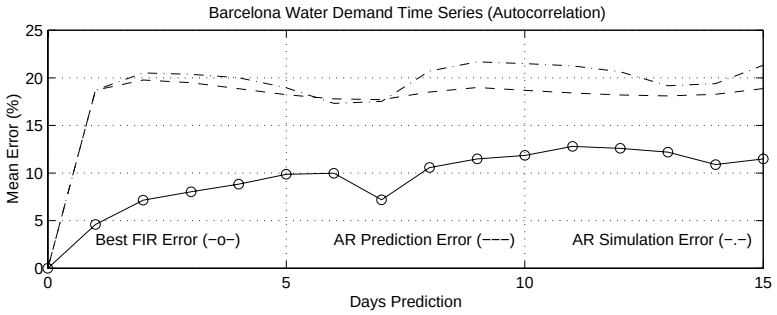

Figure 4.10: Comparative analysis of Barcelona water demand predictions: Autocorrelation *vs.* FIR.

FIR Weight Method

The idea behind the FIR weight method had been explained earlier in this chapter. Also this method can be implemented either in a simulation mode or in a prediction mode.

In the *simulation mode*, the prediction at time t is a linear combination of the values obtained for the previous seven days. The resulting model is:

$$\begin{aligned} y_t &= 0.2844 \cdot y_{t-1} + 0.0530 \cdot y_{t-2} + 0.0330 \cdot y_{t-3} \\ &\quad +0.0210 \cdot y_{t-4} + 0.0341 \cdot y_{t-5} + 0.1590 \cdot y_{t-6} \\ &\quad +0.4155 \cdot y_{t-7} \end{aligned} \tag{4.58}$$

Here, all coefficients have to be positive, because the mask qualities are always positive quantities.

There is certainly one pole at $z = 1.0$, since the sum of all coefficients has been normalized to 1.0. The pole locations are:

$$r = \begin{pmatrix} 1.0000 \\ 0.5798 + 0.7142 \cdot \mathrm{i} \\ 0.5798 - 0.7142 \cdot \mathrm{i} \\ -0.1991 + 0.8338 \cdot \mathrm{i} \\ -0.1991 - 0.8338 \cdot \mathrm{i} \\ -0.7386 + 0.3500 \cdot \mathrm{i} \\ -0.7386 - 0.3500 \cdot \mathrm{i} \end{pmatrix} \tag{4.59}$$

with the absolute values:

$$r_{\text{abs}} = \begin{pmatrix} 1.0000 \\ 0.9199 \\ 0.9199 \\ 0.8573 \\ 0.8573 \\ 0.8173 \\ 0.8173 \end{pmatrix} \tag{4.60}$$

All poles are inside the unit circle except for the single marginally–stable pole at $z = 1.0$. Multi–step predictions are obtained by reusing previous predictions in the process of making new predictions.

In the *prediction mode*, FIR weights from earlier days are being used.

Figure 4.11 compares the average errors with those of the FIR prediction. Just as in the previous two cases, the *prediction mode* outperforms the *simulation mode* because of data contamination problems. Although the errors remain below 20% most of the time, the results are worse than those obtained using least squares.

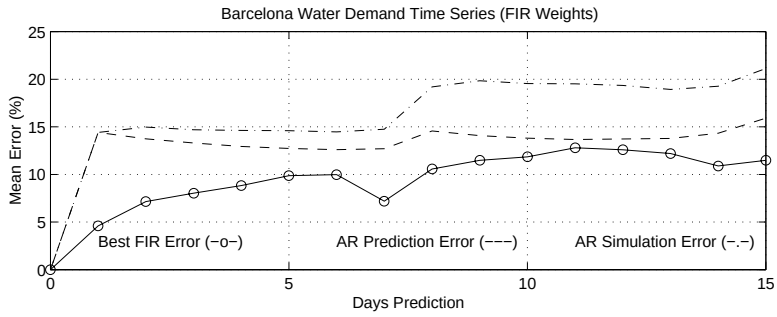

Figure 4.11: Comparative analysis of Barcelona water demand predictions: Fir weights *vs.* FIR.

Figure 4.12 compares the three AR prediction modes with each other. It is interesting to notice that the FIR weight technique consistently outperforms the autocorrelation method. The reason is that the mask qualities are better measures of the non–linear correlations between different time instants of this time series than the autocorrelation values. However, the straight–forward least squares approximation is still slightly superior.

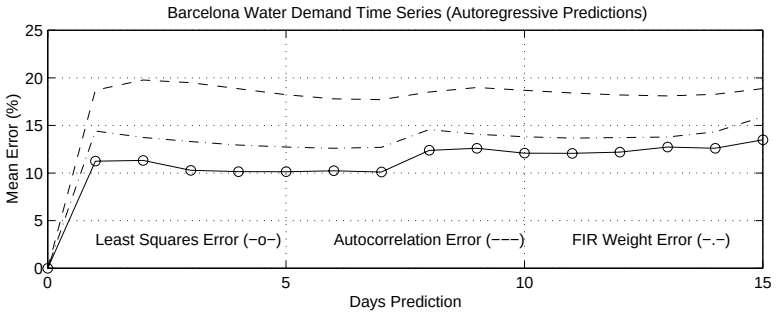

Figure 4.12: Comparative analysis of the three AR prediction models for the Barcelona water demand.

4.8.3 FIR Qualitative Prediction

Neither of the AR techniques could measure up to the performance of the *FIR qualitative simulation* in spite of the fact that also FIR has to fight against data contamination problems. However, it is perfectly feasible to also program FIR as a prediction code rather than a simulation code.

The single–step prediction proceeds as before. However, for convenience (explanation follows), an embedding dimension of 21 was chosen instead of 14. The mask candidate matrix and the resulting mask are:

$$mcan = \begin{matrix} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-2\delta t \\ t-\delta t \\ t \end{matrix} \begin{matrix} y \\ \begin{pmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ +1 \end{pmatrix} \end{matrix} \quad ; \quad mask = \begin{matrix} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-8\delta t \\ t-7\delta t \\ t-6\delta t \\ \cdots \\ t-2\delta t \\ t-\delta t \\ t \end{matrix} \begin{matrix} y \\ \begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ 0 \\ -3 \\ +1 \end{pmatrix} \end{matrix} \tag{4.61}$$

The *mask candidate matrix*, *mcan* indicates by -1 elements, where there are *potential* m–inputs, whereas the *mask* indicates by negative elements where there are the *actual* m–inputs. Evidently, FIR liked the idea of an enhanced embedding dimension and chose a deeper mask.

For a two–step prediction, the value at time $(t-\delta t)$ cannot be used for predicting the value at time t, because it is not yet known. In FIR, the corresponding entry must be masked out in the mask candidate matrix:

$$mcan = \begin{array}{c} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \begin{pmatrix} -1 \\ -1 \\ -1 \\ -1 \\ 0 \\ +1 \end{pmatrix} \end{array} \quad ; \quad mask = \begin{array}{c} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-8\delta t \\ t-7\delta t \\ t-6\delta t \\ \cdots \\ t-3\delta t \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ 0 \\ -3 \\ 0 \\ +1 \end{pmatrix} \end{array} \tag{4.62}$$

The proposed mask is reasonable. FIR chooses the value two steps back and one week back as the major elements to base the prediction upon.

For a three–step prediction, the another element must be masked out in the mask candidate matrix :

$$mcan = \begin{array}{c} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-3\delta t \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \begin{pmatrix} -1 \\ -1 \\ -1 \\ -1 \\ 0 \\ 0 \\ +1 \end{pmatrix} \end{array} \quad ; \quad mask = \begin{array}{c} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-17\delta t \\ t-16\delta t \\ t-15\delta t \\ \cdots \\ t-8\delta t \\ t-7\delta t \\ t-6\delta t \\ \cdots \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ 0 \\ -3 \\ 0 \\ 0 \\ 0 \\ +1 \end{pmatrix} \end{array} \tag{4.63}$$

Because of the low correlation with the value three steps back, FIR decided to ignore this value and choose the values one week back, 16 days back, and 21 days back. The choice of the data point at time $(t - 16\delta t)$ is hard to explain, but FIR usually knows what it is doing. This seems to be the best mask for predicting $y(t)$ given that $y(t-\delta t)$ and $y(t-2\delta t)$ may not be used.

Since nothing prevents this mask from being selected for the four–, five–, six–, and seven–day predictions, the same mask will be used in all these cases.

For the eight–day prediction, the following mask was found:

$$mcan = \begin{array}{l} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-8\delta t \\ t-7\delta t \\ \cdots \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \left(\begin{array}{r} -1 \\ -1 \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \\ +1 \end{array}\right) \end{array} \quad ; \quad mask = \begin{array}{l} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-15\delta t \\ t-14\delta t \\ t-13\delta t \\ \cdots \\ t-10\delta t \\ t-9\delta t \\ t-8\delta t \\ \cdots \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \left(\begin{array}{r} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ 0 \\ -3 \\ 0 \\ 0 \\ 0 \\ +1 \end{array}\right) \end{array} \tag{4.64}$$

Since the data point one week ago can no longer be used, FIR selects the data point two weeks ago as one of its m–inputs. However, now the data point at time $(t-16\delta t)$ is no longer that attractive, because it is too close to the one at $(t-14\delta t)$. Instead, it uses the data point at $(t-9\delta t)$. However, it is hard to explain why FIR did not choose the data point at $(t-8\delta t)$, which would have been available as well.

Evidently, the same mask will be selected for a nine–day prediction.

For a ten–day prediction, FIR chooses the following mask:

$$mcan = \begin{array}{l} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-10\delta t \\ t-9\delta t \\ \cdots \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \left(\begin{array}{r} -1 \\ -1 \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \\ +1 \end{array}\right) \end{array} \quad ; \quad mask = \begin{array}{l} \\ t-21\delta t \\ t-20\delta t \\ \cdots \\ t-17\delta t \\ t-16\delta t \\ t-15\delta t \\ t-14\delta t \\ t-13\delta t \\ \cdots \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \left(\begin{array}{r} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ -3 \\ 0 \\ 0 \\ 0 \\ +1 \end{array}\right) \end{array} \tag{4.65}$$

The youngest data point at time $(t - 10\delta t)$ is not attractive because of low correlation, and therefore, FIR chooses again the value at $(t - 16\delta t)$ as an additional m–input.

This mask is acceptable also for 11–, 12–, 13–, and 14–day predictions. Only the 15–day prediction uses again a different mask:

$$mcan = \begin{matrix} \\ t - 21\delta t \\ t - 20\delta t \\ \cdots \\ t - 15\delta t \\ t - 14\delta t \\ \cdots \\ t - \delta t \\ t \end{matrix} \begin{matrix} y \\ \begin{pmatrix} -1 \\ -1 \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \\ +1 \end{pmatrix} \end{matrix} \quad ; \quad mask = \begin{matrix} \\ t - 21\delta t \\ t - 20\delta t \\ t - 19\delta t \\ t - 18\delta t \\ t - 17\delta t \\ t - 16\delta t \\ t - 15\delta t \\ t - 14\delta t \\ \cdots \\ t - \delta t \\ t \end{matrix} \begin{matrix} y \\ \begin{pmatrix} -1 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ -3 \\ 0 \\ 0 \\ 0 \\ +1 \end{pmatrix} \end{matrix} \tag{4.66}$$

Notice that the embedding dimension was chosen identical for all masks, i.e., the mask could not have selected any values earlier than $(t - 21\delta t)$. It was necessary to enhance the embedding dimension to 21, in order to provide enough -1 elements in the mask candidate matrices for multi–step predictions.

Notice also that the complexity of the mask (the maximum number of non–zero elements) was limited to 5 in the optimal mask search, i.e., no mask could have used more than four m–inputs, but this was not a problem, as FIR consistently preferred masks of complexity 4, i.e., masks with three m–inputs.

The forecasting now proceeds exactly as before, however, in multi–step predictions, the masks are switched in accordance with the above proposed scheme, in order to prevent FIR from using already contaminated data as m–inputs.

Figure 4.13 compares the *FIR qualitative prediction* with the previously used *FIR qualitative simulation*. If the lesson learned from the three AR models extends to FIR as well, the errors would be expected to be yet smaller.

It was not to be. Although the FIR prediction mode generates results that are still better than those obtained using many of the other approaches, the FIR simulation method fared significantly better. Because FIR produces better predictions, the method is less vulnerable to data contamination than

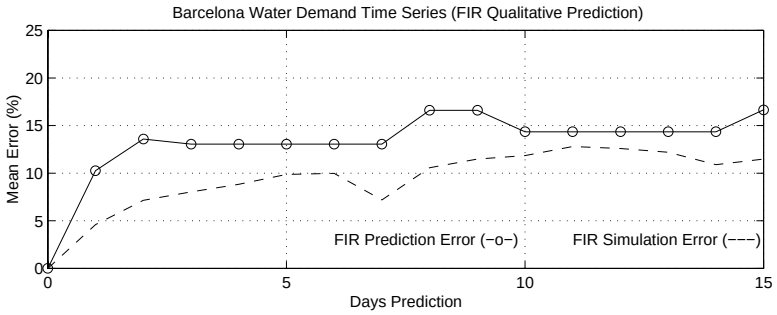

Figure 4.13: Comparison of FIR prediction and simulation for Barcelona water demand.

the other approaches. Here, the larger prediction horizon apparently was too dire a price to pay[2].

4.8.4 ARIMA Predictions

Using the approach described earlier in this chapter, the following ARIMA model was derived for Series B:

$$Y(z) = \frac{0.9460 \cdot z^7}{((z^7 - 0.5778 \cdot z^6 - 0.2093) \cdot (z^7 - 1)} \cdot E(z) \tag{4.67}$$

taking into account the weekly cycle of Series B.

Analyzing the stability of the method, it can be found that the denominator polynomial to the right, $(z^7 - 1)$, has seven roots that are all marginally stable, equidistantly spaced around the unit circle of the complex z–plane. The denominator polynomial to the left, $(z^7 - 0.5778 \cdot z^6 - 0.2093)$, has the following seven roots:

$$r = \begin{pmatrix} 0.9209 \\ 0.5916 + 0.5962 \cdot \mathrm{i} \\ 0.5916 - 0.5962 \cdot \mathrm{i} \\ -0.1068 + 0.7602 \cdot \mathrm{i} \\ -0.1068 - 0.7602 \cdot \mathrm{i} \\ -0.6563 + 0.3406 \cdot \mathrm{i} \\ -0.6563 - 0.3406 \cdot \mathrm{i} \end{pmatrix} \tag{4.68}$$

[2]The comparison, as presented here, is unfairly biased in favor of the FIR qualitative simulation. Details of why this is the case will be provided in Chapter 6 of this dissertation, where the issue will be resumed.

with the absolute values:

$$r_{\text{abs}} = \begin{pmatrix} 0.9209 \\ 0.8399 \\ 0.8399 \\ 0.7676 \\ 0.7676 \\ 0.7395 \\ 0.7395 \end{pmatrix} \tag{4.69}$$

Hence the method is indeed marginally stable, as desired.

The model is implemented as follows. Starting with the following initial errors:

$$\begin{aligned} e_{t-1} &= y_{t-1} - y_{t-8} \\ e_{t-2} &= y_{t-2} - y_{t-9} \\ e_{t-3} &= y_{t-3} - y_{t-10} \\ e_{t-4} &= y_{t-4} - y_{t-11} \\ e_{t-5} &= y_{t-5} - y_{t-12} \\ e_{t-6} &= y_{t-6} - y_{t-13} \\ e_{t-7} &= y_{t-7} - y_{t-14} \end{aligned} \tag{4.70}$$

the following recursive formulae are implemented:

$$\begin{aligned} e_t &= 0.9460 \cdot e_{t-7} \\ y_t &= 0.5778 \cdot y_{t-1} + 1.2093 \cdot y_{t-7} - 0.5778 \cdot y_{t-8} \\ &\quad -0.2093 \cdot y_{t-14} + e_t \end{aligned} \tag{4.71}$$

This method finally led to decent results.

Figure 4.14 compares the ARIMA simulation with the previously obtained FIR qualitative simulation.

FIR outperformed also the ARIMA prediction, but not by much. The ARIMA prediction is rather decent. Why did this methodology perform better than other linear prediction methods? The answer is simple: ARIMA is the only method that makes explicit use of the time-varying nature of Series B. It exploits explicitly the weekly cycle. None of the other techniques (including FIR) paid any attention to this piece of information.

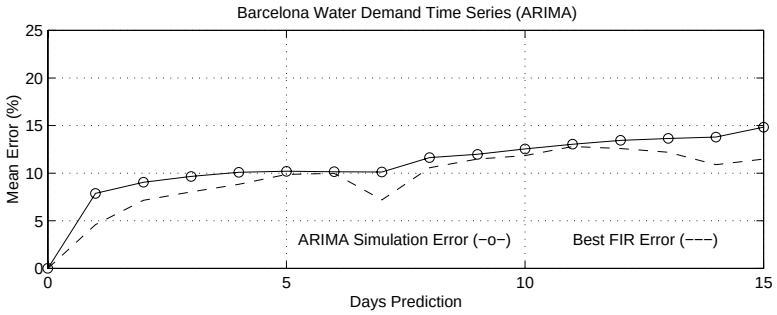

Figure 4.14: Comparison of ARIMA and FIR simulations for Barcelona water demand.

Figure 4.15 shows the single–day, eight–day, and 15–day predictions obtained using the ARIMA model. Contrary to FIR, ARIMA does not filter out noise, i.e., the standard deviation does not get reduced in multi–step predictions.

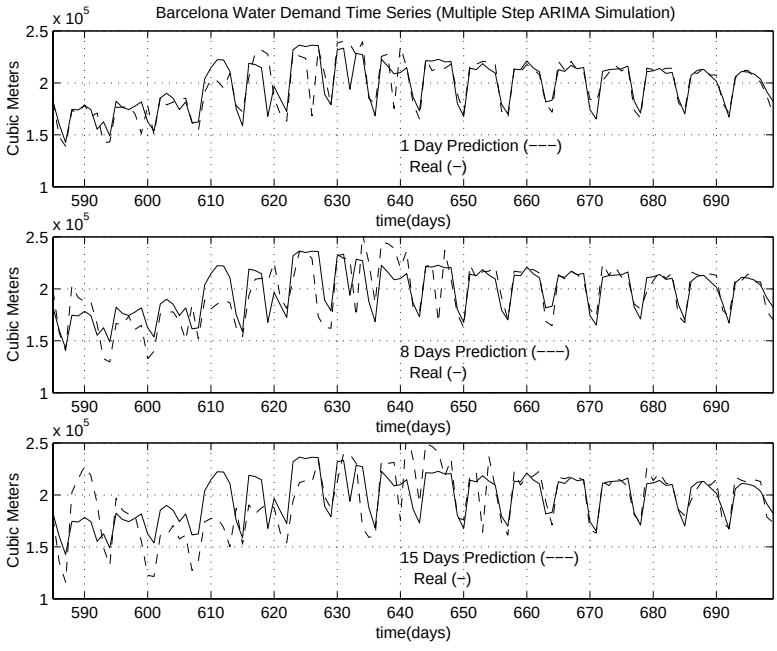

Figure 4.15: Multi–day predictions of Barcelona water demand using ARIMA model.

In (Quevedo *et al.* 1988), an improved ARIMA (Box–Jenkins) model was proposed for the same system. That model used interventions to account for the effects of holidays on water consumption, i.e., it paid attention to the

vacation month of August as well as the most important public holidays in Barcelona, such as Easter and Christmas. This model was able to reduce the one–day prediction error to about the same levels as FIR. No multiple–day predictions were made at that time, because the customer, *Aigües de Barcelona*, did not require predictions beyond one day.

This model, which is considered the most important reference for comparison of the forecasting results of the Barcelona water demand series, is still in use today in the city's water distribution management system. The model was able to reduce the one–day prediction error significantly as compared to the simpler ARIMA model presented in this thesis.

Constructing this sophisticated ARIMA model was a rather elaborate task. The interventions applied to the series by this model are specific to this series, and required a minute understanding and analysis of the phenomena that dictate the water consumption in Barcelona.

In contrast, the FIR model could be set up within a few hours, and doing so did not require any analysis of the time series at all. The five–nearest–neighbor rule provides FIR with a feature that is somehow similar in its effects to the interventions of the ARIMA model, at least for holidays that extend beyond a single day. The fact that FIR was able to produce forecasting errors that are not significantly different from those obtained by the much more sophisticated ARIMA model, speaks for FIR's ability to extract most of the information provided in the training data set automatically and reliably.

4.8.5 NAR Predictions

Why did FIR outperform even the ARIMA predictions introduced in the previous section? Maybe, because the system is non–linear, and all estimators that were introduced so far, with the exception of FIR, were linear estimators. In the sequel, two non–linear estimators shall be introduced.

Both the aggregation method and the refinement method introduced earlier were used to come up with the best NAR model for Series B. The resulting model uses the following recursion formula:

$$\begin{aligned} y_t &= 0.6510 \cdot y_{t-1} + 0.4747 \cdot y_{t-7} - 0.3036 \cdot y_{t-8} + 0.3128 \cdot y_{t-14} \quad (4.72) \\ &\quad -0.1763 \cdot 10^{-6} \cdot y_{t-9}^2 - 0.5277 \cdot 10^{-6} \cdot y_{t-15}^2 \end{aligned}$$

The quadratic terms have small coefficients, but they are nevertheless important, since they get multiplied with y^2, rather than y, where $\|y(t)\| \approx 2 \cdot 10^5$ m^3.

Figure 4.16 compares the NAR simulation with the previously obtained ARIMA and FIR simulations.

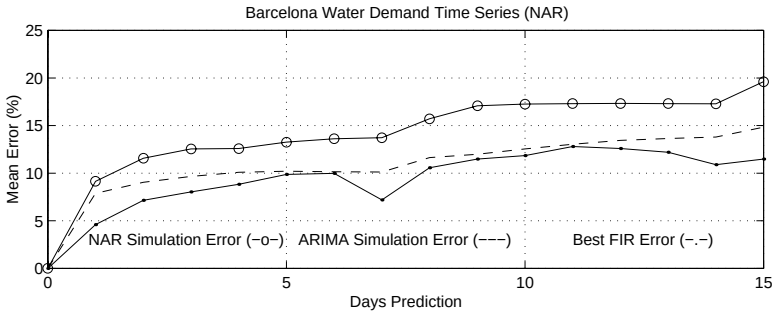

Figure 4.16: Comparison of NAR, ARIMA, and FIR simulations for Barcelona water demand.

The results are not as good as those obtained for either the ARIMA or the FIR model. Figure 4.17 shows the single–day, eight–day, and 15–day predictions obtained using the NAR model. Although there was no guarantee for stability, the predictions seem to be stable. Yet, and this is one of the major drawbacks of the NAR methodology, there is no guarantee that the predictions will always remain stable.

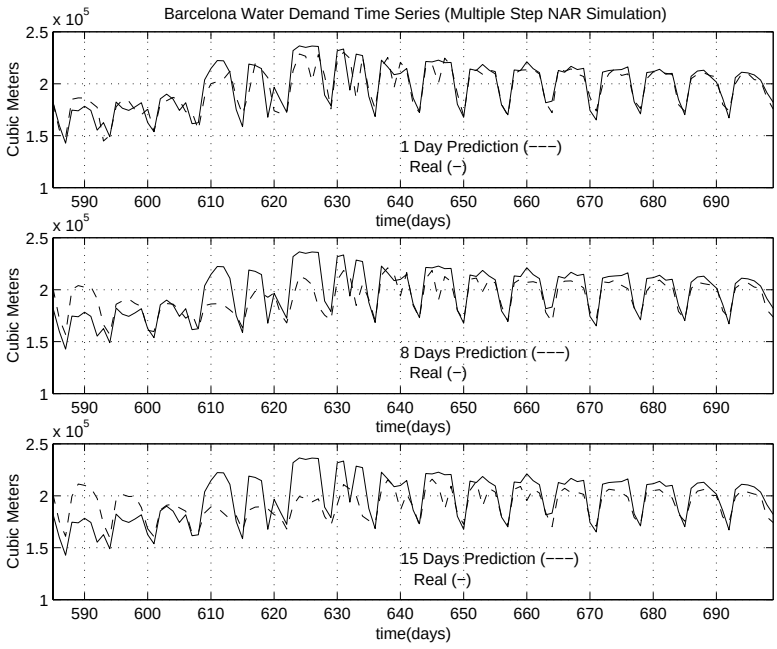

Figure 4.17: Multi–day predictions of Barcelona water demand using NAR model.

4.8.6 ANN Predictions

Earlier investigations have shown that FIR and ANN are two comparative techniques in the sense that FIR usually works well, when ANN performs well, and vice–versa. Thus, it makes sense to also generate an ANN model of the same series for the purpose of comparing it with the other models.

The ANN constructed to this end is a standard feed–forward network, trained using backpropagation. The network contains 14 input nodes, receiving the signals $y_{t-1} \ldots y_{t-14}$. It contains a single output node, delivering the estimate y_t. It furthermore contains one hidden layer with 20 neurons. All neurons use sigmoidal activation functions of the hyperbolic tangent (tgh) type.

The model is based on an earlier model of identical structure (Griñó 1992). Like its predecessor, the model was both trained and simulated in NeuralWorks (NeuralWare 1993). The weights were re–identified, since full information about the earlier model was no longer available.

The multi–step prediction errors are presented in Figure 4.18, which compares the ANN simulation with the previously obtained ARIMA and FIR simulations.

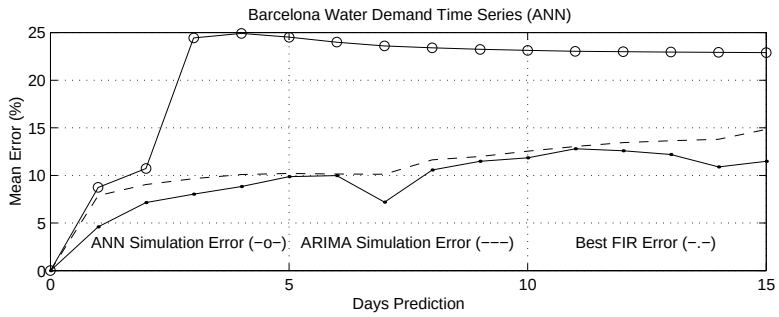

Figure 4.18: Comparison of ANN, ARIMA, and FIR simulations for Barcelona water demand.

The one–day and two–day predictions are almost as good as the ARIMA predictions, but for longer–term predictions, this ANN model performs poorly.

Figure 4.19 shows the single–day, eight–day, and 15–day predictions obtained using the ANN model. As was the case for the NAR model, there is no guarantee of stability. Contrary to the NAR model, the identified ANN model is unstable, i.e., if predictions are made over multiple days, the oscillations grow in amplitude. This is the reason for the poor performance of the proposed ANN model when applied to multiple–day predictions.

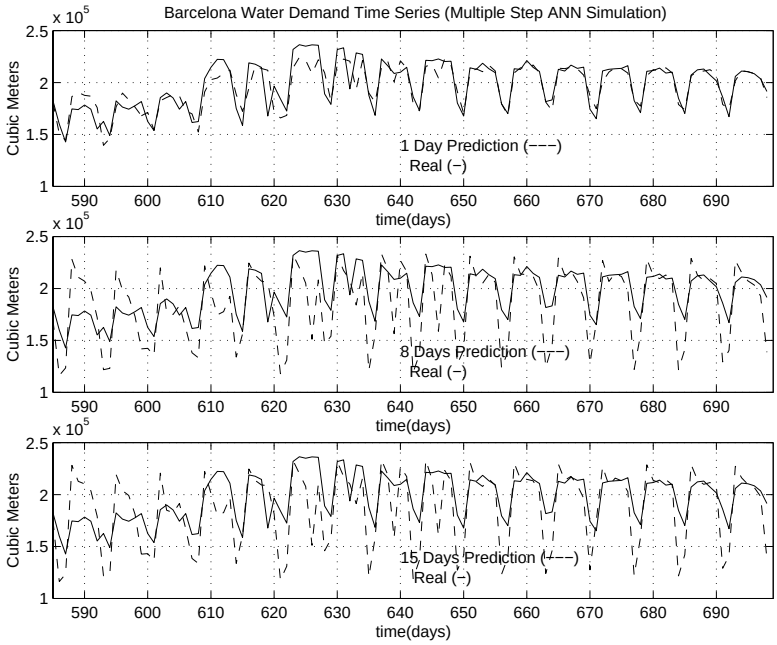

Figure 4.19: Multi–day predictions of Barcelona water demand using ANN model.

FIR is the only one among the non–linear modeling technique discussed in this chapter that is guaranteed to remain stable when used in a simulation mode. FIR can make *incorrect* predictions, but never *unstable* ones. It would not know how. FIR can only predict patterns that it has seen before.

Of course, it would be possible to program an ANN that can be used in prediction mode. To this end, one could either program a single ANN with 15 output nodes, one for each of the subsequent 15 days, or alternatively, one could program 15 separate ANNs with one output node each, each predicting a different day. In such an approach, stability would not be an issue.

There exist many different types of ANNs: feedforward *vs.* recurrent networks, static *vs.* dynamic networks (Korn 1995). Unfortunately, there does not exist any theory that could be used to decide, which type of ANN would work best in any given situation, i.e., the task of finding the best possible ANN for any given problem is a very tedious task indeed. In this chapter, only the simplest, though most widely used, type was discussed. There is no particular reason why this type of ANN should be the one best suited for the task at hand. Hence the search for competing methods has certainly not been exhaustive.

4.9 Time Series R: Rotterdam Water Demand

The second time series discussed in this chapter represents the water demand of a part of the city of Rotterdam, called the *Berenplaat* (Europoort 1986). The measurement data are shown in Figure 4.20. The total water volume contained in this system is $\|y(t)\| \approx 2.5 \cdot 10^5$ m^3, i.e., comparable in size with that contained in the Barcelona system.

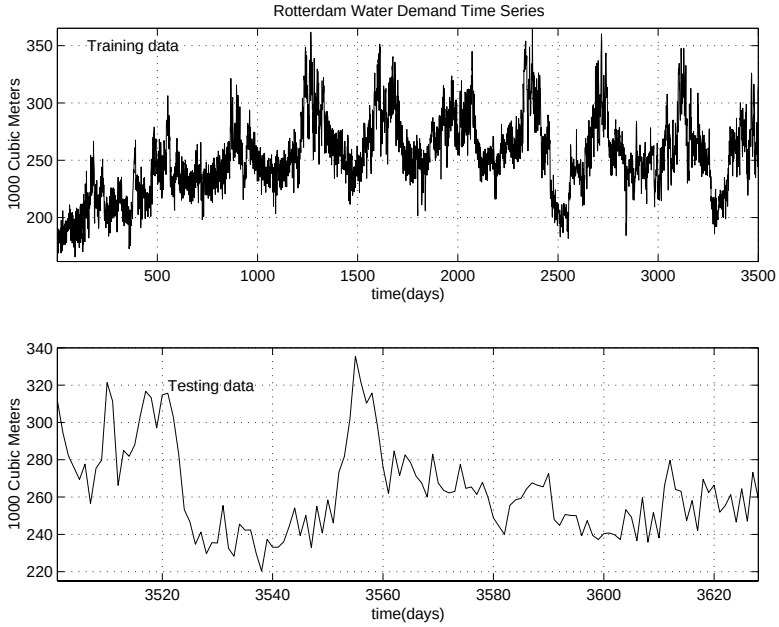

Figure 4.20: Rotterdam water demand data.

Series R can be classified as shown in Table 4.2. The behavior of Series R is considerably more stochastic than that of Series B. No explanation has been found, why this is the case. A smaller subdivision would be expected to contain more variability in the data, i.e., more "noise." Yet, the subdivision considered in the Berenplaat system is not smaller than that considered in the Barcelona system. Yet, Series R must be classified as noisy. Luckily, more measurement data were available for Rotterdam, namely 10 years worth of daily measurements, from January 1986 to December 1995.

There is quite a bit of auto–correlation contained in Series R, as shown in Figure 4.21, which makes it probable that meaningful predictions can be made. As in the case for Barcelona, also this auto–correlation function shows a weekly cycle. However, the peaks are much smaller and decay much more rapidly than in the case of the Barcelona series. There exists a significant seasonal cycle, but this may be difficult to exploit, since the peak in the

Table 4.2: Classification of Time Series R.

natural	R	synthetic	
stationary		non–stationary	R
time invariant		time varying	R
low dimensional		stochastic	R
clean		noisy	R
short		long	R
dormant		active	R
documented	R	blind	
linear		non–linear	R
scalar	R	vector	
single recording	R	multiple recordings	
continuous	R	discrete	

auto–correlation function reached after 365 days exhibits an amplitude that is only about as high as that after 50 days.

Due to the more stochastic nature of this time series, more data points were needed for model identification. Of the available 10 years of data, 9.5 years (corresponding to 3500 data points) were used as training data (i.e., for model identification), whereas the remaining 0.5 years worth of data were used as testing data (i.e., for model validation).

4.9.1 FIR Qualitative Simulation

Due to the shape of the autocorrelation function, it was decided to limit the mask depth to seven days. FIR found the following optimal mask:

$$\begin{array}{c} \\ t-7\delta t \\ t-6\delta t \\ t-5\delta t \\ t-4\delta t \\ t-3\delta t \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} y \\ \left(\begin{array}{r} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ -3 \\ +1 \end{array} \right) \end{array} \tag{4.73}$$

Again, the model that FIR proposes is quite reasonable. Because of the more rapid decay of the auto–correlation function, the data point $y(t - 14\delta t)$ is

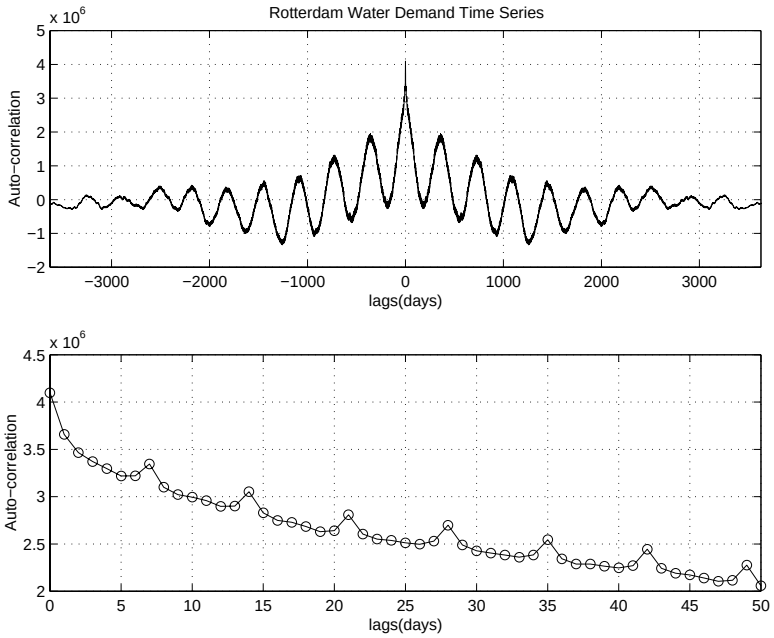

Figure 4.21: Auto–correlation of Rotterdam water demand data.

less relevant than in the previous case. Instead, FIR chose to also use the data point $y(t - 3\delta t)$ for the prediction.

Figure 4.22 shows the averaged error, $err[j]$, and the two averaged accumulated confidence functions, $c_a[j]$, as a function of the number of sampling periods, j, that the measurement data lag behind the prediction. Just as in the case of the Barcelona series, the errors are compared to those of the daily and weekly trivial predictors.

This time, the daily trivial prdictor performed better than the weekly trivial predictor, which is not surprising, taking into account the weeker weekly cycle of Series R. Both trivial predictors outperformed FIR by leaps and bounds, i.e., FIR did not predict *anything*.

Looking at the confidence values, they are about equally high for the Rotterdam series as for the Barcelona series. The reason is that Series R offers considerably more training data, i.e., FIR is able to find plenty of close neighbors in the input space. It simply happens that there is a large dispersion between predicted outputs for similar neighbors, which makes it impossible for FIR to know what to predict. The dispersion between outputs *is* punished in the confidence formula (as shall be shown in the next chapter), but evidently not enough to raise a serious flag.

Figure 4.23 compares the one–day prediction, the eight–day prediction, and the fifteen–day prediction with the measurement data.

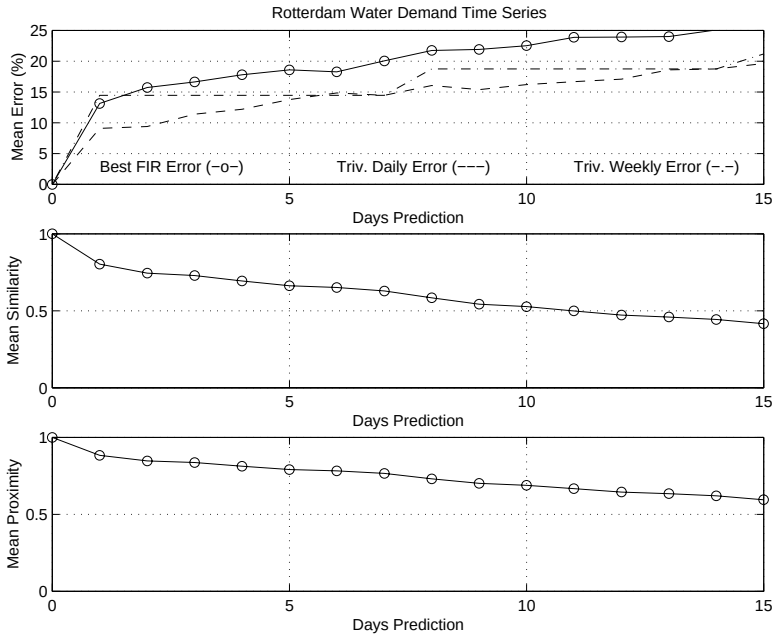

Figure 4.22: Rotterdam water demand multiple–step simulation using FIR.

Essentially, FIR is doing the same as the trivial daily predictor: its predictions lag one day, eight days, and 15 days behind, simply because the unknowable cannot be predicted.

Now that FIR has been defeated, it will be interesting to check how the contending methodologies fare in this case.

4.9.2 AR Predictions

In the sequel, FIR is being compared against three different AR models, one using least squares, the second using the autocorrelation coefficients, and the third using the FIR mask qualities.

Least Squares Method

As in the case of Series B, a simulation model and a prediction model have been computed. Figure 4.24 compares the average errors of these models with those of the FIR prediction and with the trivial daily predictor.

The prediction mode and the simulation mode work about equally well. Both outperform FIR, but neither of them reaches the "quality" of the trivial predictor, i.e., also these estimators do not accomplish *anything*. FIR outsmarts itself trying to make sense out of correlated noise.

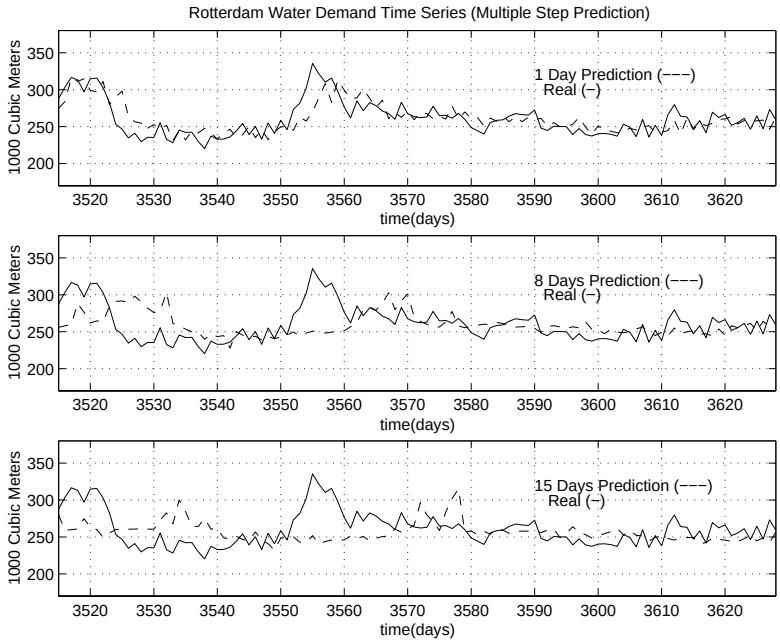

Figure 4.23: Rotterdam water demand multiple–step simulation using FIR.

Autocorrelation Method

In analogy with the Barcelona series, the auto–correlation approach shall now be tried both in simulation and in prediction mode. Figure 4.25 compares the average errors with those of the FIR prediction as well as with the trivial daily predictor.

If there is anyhting exploitable in this series beyond the correlation with the previous day, *this* method should find it. There simply is no information in this data set that can be exploited. The auto–correlation method works as well (and as poorly) as the least squares method. Both outperform FIR, but neither can measure up to the performance of the trivial predictor.

FIR Weight Method

Figure 4.26 compares the average errors of the FIR weight method with those of the FIR prediction and with those of the trivial predictor.

The performance is exactly the same as for the previous two methods. Why does FIR perform more poorly? All of the other techniques preserve the mean value and standard deviation. FIR recognizes the garbage data as noise and starts filtering the noise out, thereby effectively reducing the standard deviation. Over multiple steps, its performance approaches that of the naïve

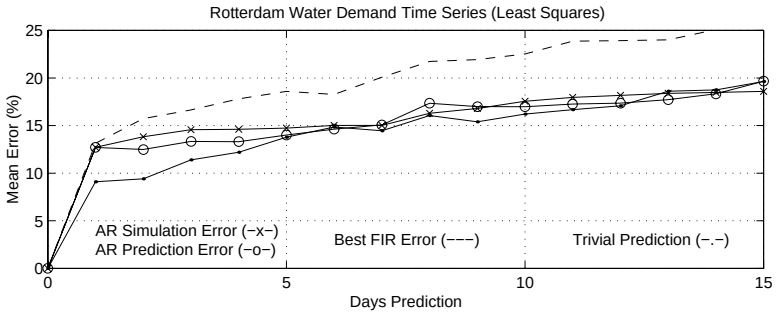

Figure 4.24: Comparative analysis of Rotterdam water demand predictions: Least squares *vs.* FIR.

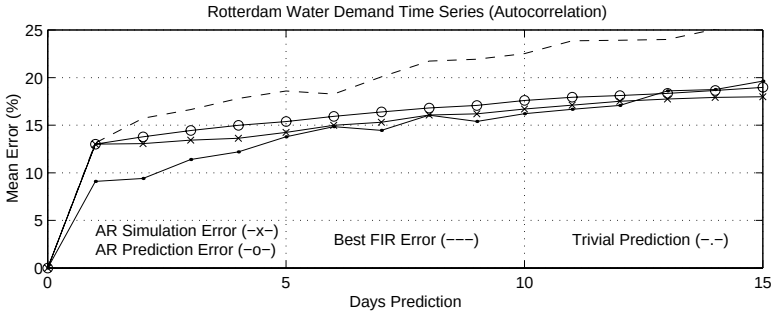

Figure 4.25: Comparative analysis of Rotterdam water demand predictions: Autocorrelation *vs.* FIR.

predictor. This is severely punished in the error formula used. Is this good or bad? The answer to this question depends on the point of view. If the unpredictable data are interpreted as valuable information, then the AR techniques indeed outperform FIR. If, on the other hand, the unpredictable nature of the data is interpreted as noise, then FIR does exactly what it is supposed to do: namely try to rid itself of the noise.

In the sequel, the more serious contenders shall be analyzed.

4.9.3 ARIMA Predictions

The following ARIMA model for Series R:

$$Y(z) = \frac{(z - 0.45) \cdot (z^7 - 0.95)}{(z - 1) \cdot (z^7 - 1)} \cdot E(z) \tag{4.74}$$

was taken from the open literature (Baggelaar 1992). This was the best ARIMA model that the researchers in Rotterdam found. As in the case of Barcelona, the model was only used for single–day predictions.

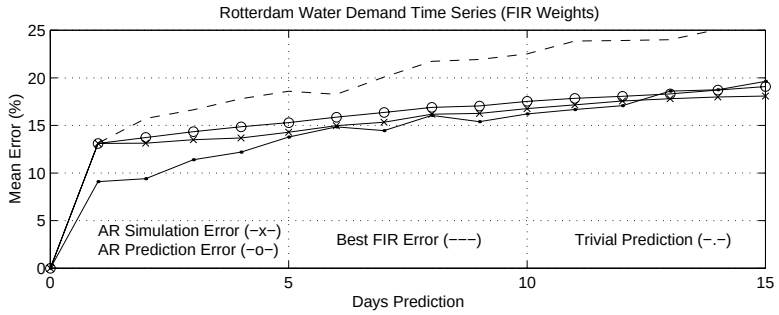

Figure 4.26: Comparative analysis of Rotterdam water demand predictions: Fir weights *vs.* FIR.

Analyzing the stability of the method, it can be seen that the method is indeed marginally stable, as desired. However, there is a double pole at $z = 1$, which can be potentially harmful, as it acts like an open integrator on the mean value of the series to be predicted, i.e., it can be expected that, over multiple–day predictions, the signals will, on average, grow linearly in magnitude, rather than staying constant.

In accordance with (Baggelaar 1992), the model was implemented using the following recursion:

$$\begin{aligned} e_t &= N(0,1) \\ y_t &= y_{t-1} + y_{t-7} - y_{t-8} \\ &\quad + e_t - 0.45 \cdot e_{t-1} - 0.95 \cdot e_{t-7} + 0.4275 \cdot e_{t-8} \end{aligned} \tag{4.75}$$

where $N(0,1)$ denotes a random number that is normally distributed with a mean value of 0.0 and a standard deviation of 1.0.

Figure 4.27 compares the ARIMA simulation with the previously obtained FIR qualitative simulation as well as the trivial predictor.

The method evidently fares about as badly as FIR. Figure 4.28 shows the single–day, eight–day, and 15–day predictions obtained using the ARIMA model.

As expected, the predictions grow in amplitude over multiple–day predictions in spite of the fact that the method is marginally stable. The model had never been intended for multiple–day predictions, i.e., this performance may be acceptable. However, even the single–day prediction is worse than the trivial prediction, at least, when using the error formula that was proposed in Chapter 3 of this dissertation.

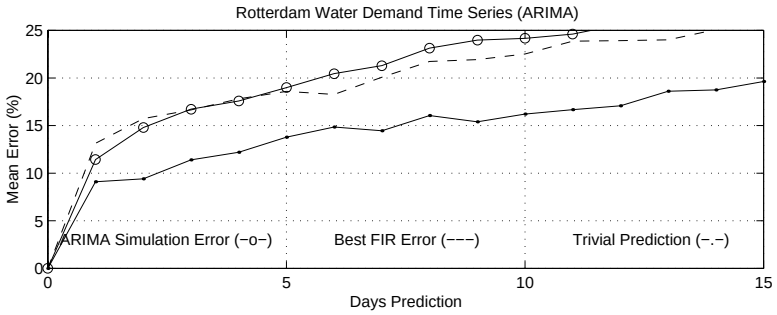

Figure 4.27: Comparison of ARIMA and FIR simulations for Rotterdam water demand.

4.9.4 NAR Predictions

Both the aggregation method and the refinement method introduced earlier were used to come up with the best NAR model for Series R. The resulting model uses the following recursion formula:

$$\begin{aligned} y_t &= 0.65753 \cdot y_{t-1} + 0.06674 \cdot y_{t-2} + 0.31 \cdot y_{t-14} \\ &\quad +0.0011723 \cdot y_{t-3} \cdot y_{t-7} - 0.0009048 \cdot y_{t-3} \cdot y_{t-8} \\ &\quad -0.0003983 \cdot y_{t-15}^2 \end{aligned} \tag{4.76}$$

Figure 4.29 compares the NAR simulation with the previously obtained FIR simulations as well as the trivial predictor.

NAR performs a little better than FIR, but not as well as the trivial predictor. Figure 4.17 shows the single–day, eight–day, and 15–day predictions obtained using the NAR model.

Similarly to FIR, the NAR model here ignores the higher frequency components of the series, i.e., the noise, and essentially predicts the mean value.

4.9.5 ANN Predictions

Finally, an ANN model was constructed. It is the same type of ANN that was used for the Barcelona data. However, the Rotterdam network only contains seven input nodes, receiving the signals $y_{t-1} \ldots y_{t-7}$. It contains a single output node, delivering the estimate y_t. It furthermore contains one hidden layer with 10 neurons. As before, all neurons use sigmoidal activation functions of the hyperbolic tangent type.

The multi–step prediction errors are presented in Figure 4.31, which compares the ANN simulation with the previously obtained FIR simulations

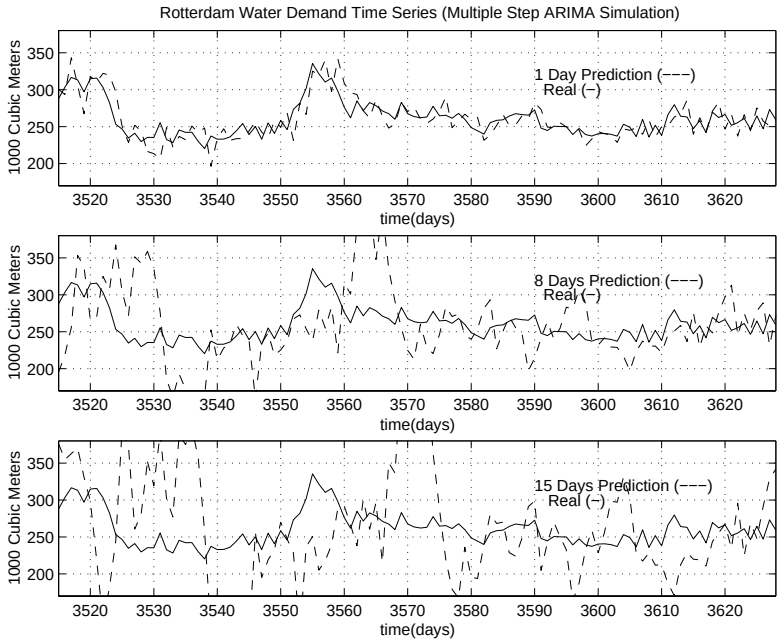

Figure 4.28: Multi–day predictions of Rotterdam water demand using ARIMA model.

as well as with the trivial predictor. The ANN fares a little better than the FIR model.

Figure 4.32 shows the single–day, eight–day, and 15–day predictions obtained using the ANN model. Although there is no guarantee of stability, this ANN model seems to be stable.

What can be concluded? None of the techniques that were applied to Series R outperformed the dayly trivial predictor. The fact that a series contains a significant amount of auto–correlation does not necessarily make it predictable beyond what the trivial predictor can accomplish. FIR performed a little worse than all other techniques, in terms of the chosen error function, because it considers the high–frequency behavior of Series R to be noise and correspondingly filters it out.

Looking at the training data, a strong seasonal dependency can be observed. Clearly, it should be possible to predict the seasonal variation. However, to this end, a different set of experiments is needed. The series is seriously oversampled for the purpose of predicting seasonal variations. In terms of FIR terminology, a mask of considerably larger depth would be needed in order to predict the seasonal variations. Yet, such a mask would not help at all for the purpose of predicting the daily water consumption. There simply is not enough information contained in the data to predict the

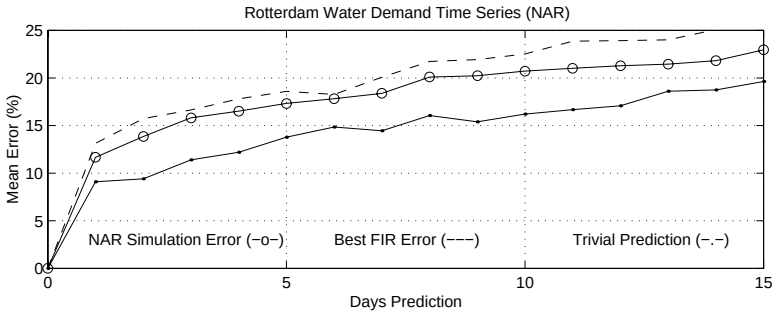

Figure 4.29: Comparison of NAR and FIR simulations for Rotterdam water demand.

daily consumption better than using the trivial predictor.

Overall, FIR is the most robust of all the techniques applied. It exploits non–linear system behavior confidently and reliably, without ever turning unstable. It is less affected than most other techniques by mildly non–stationary system behavior, and the method can be applied very easily. Constructing a FIR model is not more complicated than constructing an AR model.

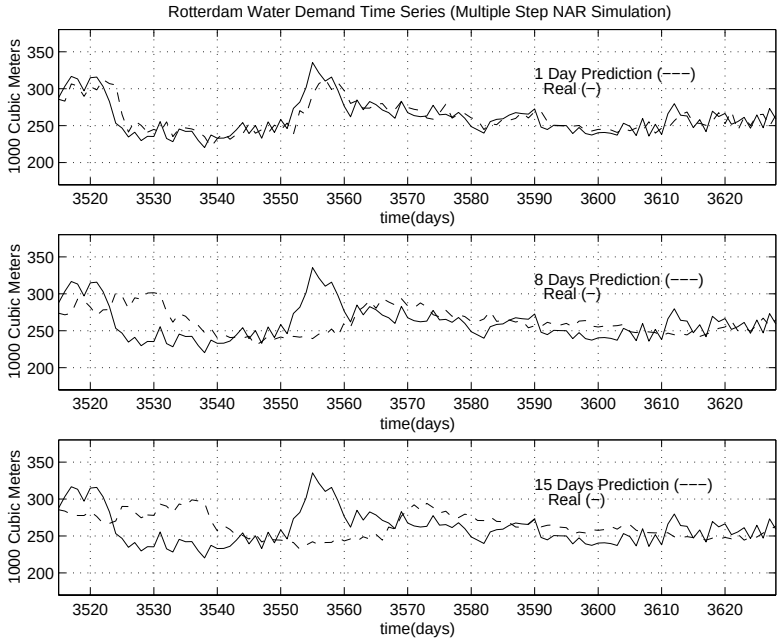

Figure 4.30: Multi–day predictions of Rotterdam water demand using NAR model.

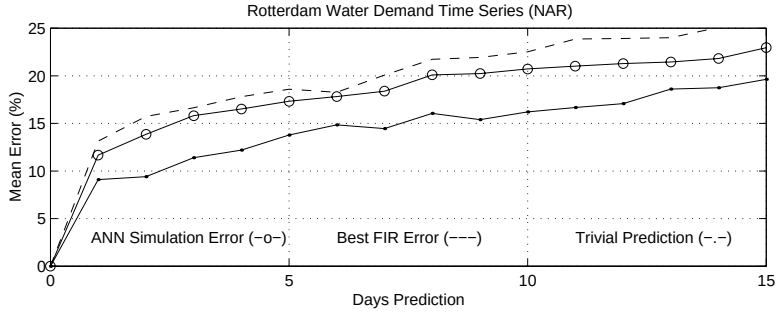

Figure 4.31: Comparison of ANN and FIR simulations for Rotterdam water demand.

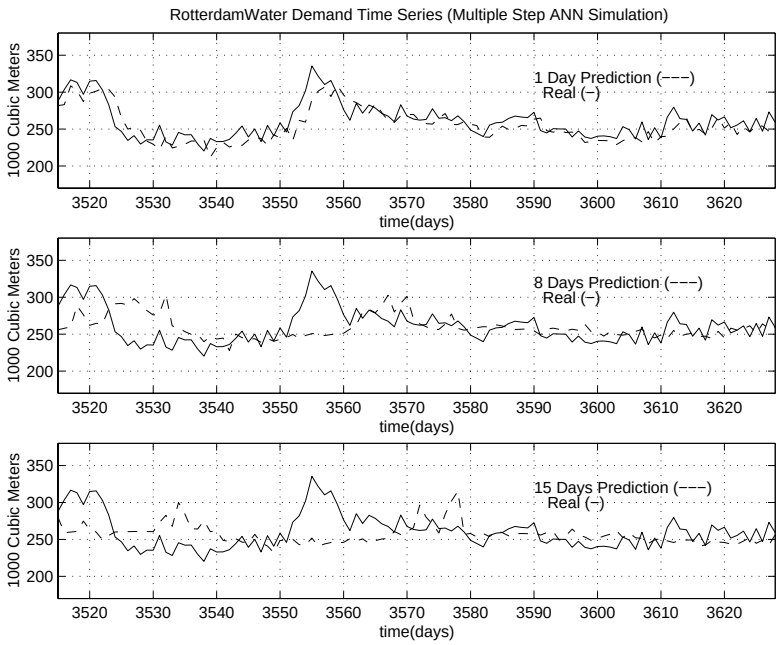

Figure 4.32: Multi–day predictions of Rotterdam water demand using ANN model.

Chapter 5

Confidence Measures for Predictions in Fuzzy Inductive Reasoning

5.1 Introduction

Models never reflect all facets of reality. They are always reductionistic in nature, and consequently, simulation results are never totally reliable. Hence it is important to always interpret simulation results with caution and a certain degree of scepticism.

The degree of uncertainty associated with a model of a system depends heavily on the nature of that system. Simple man–made engineering systems, such as electronic circuits, are characterized by a small degree of uncertainty, since it is an actual design goal when producing these systems to keep the degree of uncertainty small. On the other hand, biological or economic systems are usually characterized by a fairly large degree of uncertainty.

Although the request for scepticism is a good mandate on moral grounds, it is doubtful whether such a demand is also practical. How should, for example, medical practitioners know how to judge the reliability of a prediction made about the status of one of their patients? They have no way of assessing the reliability of a prediction made by an obscure simulation model that is driven by measurement data taken from the patient. In all likelihood, the model underlying this simulation was developed by someone else, and they may not even know how it works. All they know is how to interpret the results that come out of the computer. Hence it is important to instil scepticism into the simulation software itself, rather than demanding it of its users.

Assessing the inaccuracy of a simulation result is in itself a modeling task. Yet, the same methodology that is used to model the output to be predicted cannot be used to model its error. This would lead to a paradoxical situation. If it indeed were possible to compute, in a deterministic sense, the inaccuracy of a prediction made, then one could simply subtract the predicted prediction error from the prediction itself and obtain the precise value of the output. Evidently, this cannot be done. The modeling error can only be modeled in a statistical sense.

In this chapter, two confidence measures implemented inside the fuzzy inductive reasoning methodology will be described that assess the error of a prediction made simultaneously with making the prediction.

In a robust modeling methodology capable of dealing with model uncertainty (as qualitative modeling techniques should always be), modeling the modeling error should not be an afterthought. Modeling the output and modeling its error should be done simultaneously. A modeling and simulation methodology that does not take the model uncertainty into consideration from the beginning is not robust when dealing with uncertain situations.

In the next section, the problems behind making decisions under uncertainty are illuminated. In the subsequent sections, the two confidence measures, a *proximity measure* and a *similarity measure*, are described in the context of the Fuzzy Inductive Reasoning (FIR) methodology.

The chapter ends with an analysis of the confidence measures obtained for two time series, which help to discuss the effectiveness of the two proposed confidence measures.

A shorter version of this chapter has meanwhile been accepted for publication in a special issue on FIR of the *International Journal of General Systems* (Cellier *et al.* 1998).

5.2 Decision Making Under Uncertainty

In Chapter 3, it was shown that FIR uses a normalized defuzzification, the *position value*, to determine the five nearest neighbors of a new input record in the experience data base. The same information is also being used to quantify the relative importance of these neighbors in interpolating the position value of the data point to be predicted in the output space.

Normalization is important, because the different variables making up a record in a multivariate time series may represent different physical variables that can be of vastly different magnitudes. For example, the first m–input, i_1, may represent yesterday's water consumption of a rural area of Catalunya,

whereas i_2 may represent yesterday's ambient temperature in the same area, assuming that the water consumption depends on the temperature, because orchards will need to be watered only during summer months, i.e., when it is hot. If i_1 is measured in m^3/day and i_2 is measured in ^oC, the numerical values of the two variables will differ by several orders of magnitude. In order to be able to compare them to each other, these variables need to be normalized.

The position values, p_i, used by the FIR methodology map each variable into the range $[1, n_{\text{cl}}]$, where n_{cl} denotes the number of classes associated with the fuzzification of the given variable, usually a number between three and five. Thus, if i_1 and i_2 are both fuzzified into three classes, $\|p_1\| \approx \|p_2\| \approx 2.0$, although $\|i_1\| \approx 2 \cdot 10^5$ and $\|i_2\| \approx 20.0$.

Figure 5.1 explains the mapping of the input space to the output space. Each m–input, i_j, is represented by its position value p_j.

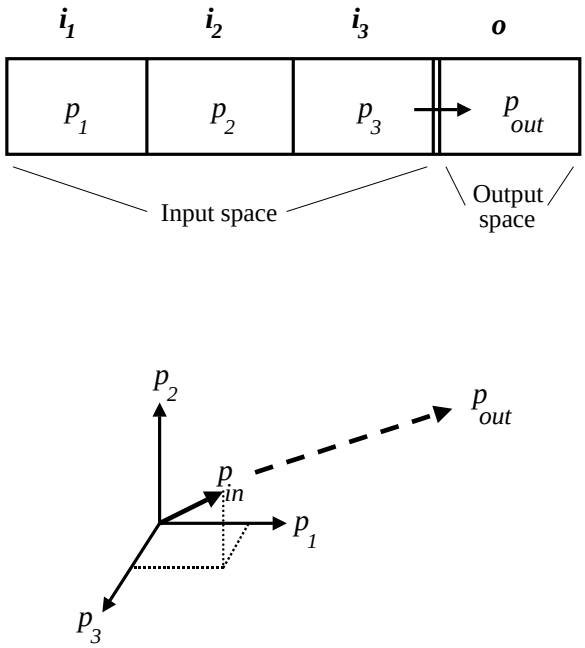

Figure 5.1: Mapping of input space to output space.

The variable p_j represents the position value of the j^{th} m–input. It also represents an independent variable (an axis) in the n–dimensional input

space, where n is the number of m–inputs. The individual p_j values are concatenated to form the input position vector, $\mathbf{p_{in}}$:

$$\mathbf{p_{in}} = [p_1, p_2, \ldots, p_n] \tag{5.1}$$

Thus, $\mathbf{p_{in}}$ is a vector in the n–dimensional input space.

The variable p_{out} represents the scalar output position value in the one–dimensional output space. The experience data base is a fuzzy metric that maps input positions in the vector input space to output positions in the scalar output space.

The map is *incomplete*, since the experience data base does not contain records for every possible input position, and it may be *inconsistent*, since very similar or even identical input positions can be mapped into quite different output positions.

A FIR prediction consists of comparing a new input position with its five nearest neighbors in the experience data base (the training data set), and making a prediction about the corresponding output position by interpolating among the output positions of the five nearest neighbors in the output space. This is a process of *decision making under uncertainty* because of the incompleteness and potential inconsistency of the records contained in the training data set.

Because of this uncertainty, a *confidence value* should be associated with every prediction made. The confidence value needs to take into account the *quantity of information* contained in the training data set, i.e., the degree of completeness, as well as its *quality*, i.e., the degree of consistency.

Table 5.1 explains these two aspects in more detail.

Table 5.1: Decision making under uncertainty.

Quantity of Information	Experience	Ignorance
Quality of Information	Competence	Confusion
Decision	Confidence	Insecurity

The *quantity of information* denotes the amount of knowledge (or experience) contained in the experience data base. It is directly related to the number of data records contained in the training data set and also to the richness in excitation.

Figure 5.2 shows the dispersion among the five nearest neighbors in the input space. The new input vector is represented by a square box (actually an n–dimensional hypercube) drawn into the n–dimensional input space. The

five nearest neighbors are represented by circles (n–dimensional hyperglobes) drawn into the same space.

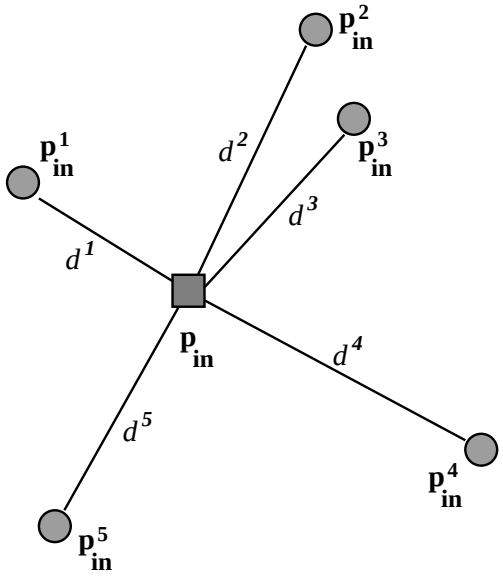

Figure 5.2: Dispersion among neighbors in input space.

The variable d^j denotes the "distance" of the j^{th} neighbor from the new input record. The "distance" value can denote a Euclidean distance or any other suitable norm:

$$d^j = \|\mathbf{p_{in}} - \mathbf{p_{in}}^j\| \tag{5.2}$$

The dispersion among the five nearest neighbors in the input space can be used as a measure of the quantity of information. For example, when predicting uncorrelated white noise (an impossible task), the dispersion among the neighbors in the input space can be fully controlled by adding more and more data records to the training data set. Thus, it is possible to gather as much information as one wants about uncorrelated white noise.

Yet, the prediction will not contain any information, because the *quality of information* is nil, since the dispersion of ouput positions in the output space for the five nearest neighbors will cover the entire range of possible

values from 1.0 to n_{cl}. This is shown in Figure 5.3, where the maps for the five nearest neighbors from the input space to the output space are shown.

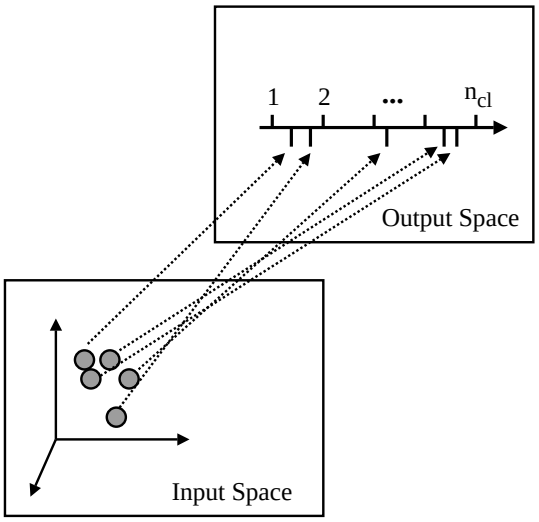

Figure 5.3: Dispersion among neighbors in output space.

Although the five nearest neighbors are close to each other in the input space (small dispersion in the input space), the corresponding observed output positions cover the entire range of values between 1.0 and n_{cl} (large dispersion in the output space), i.e., making a prediction is like throwing a dice, which of course, is the best that can be done when predicting uncorrelated white noise.

In the sequel, two different *quality measures* are presented that estimate the confidence associated with a prediction. One is based on the concept of *proximity*, the other is based on the concept of *similarity* with and among the five nearest neighbors in the training data set.

5.3 The Proximity Measure

The idea behind assessing the reliability of a prediction by means of a proximity measure is related to establishing distance measures between the testing input state and the training input states of its five nearest neighbors in the experience data base, and to establishing distance measures between the output states of the five nearest neighbors among themselves.

The *average distance* used to determine the input confidence measure is computed as a weighted sum of the relative distances of the five nearest neighbors in the input space:

$$d_{\text{conf}_{\text{in}}} = \sum_{j=1}^{5} w_{\text{rel}}^{j} \cdot d^{j} \tag{5.3}$$

where w_{rel}^{j} are the relative weights, established in Chapter 3 of this thesis, that are themselves functions of the distances d^{j} between the new input position and its j^{th} nearest neighbor.

The largest possible input distance value can be calculated as:

$$d_{\text{conf}_{\text{in}_{\text{max}}}} = \sqrt{\sum_{i=1}^{n} (n_i - 1)^2} \tag{5.4}$$

where n_i is the number of classes used in the fuzzification of the i^{th} input variable, assuming that a Euclidean norm is used in the computation of distances.

Consequently, the confidence value related to the proximity of the five nearest neighbors in the input space can be defined as:

$$conf_{\text{prox}_{\text{in}}} = 1.0 - \frac{d_{\text{conf}_{\text{in}}}}{d_{\text{conf}_{\text{in}_{\text{max}}}}} \tag{5.5}$$

where $conf_{\text{prox}_{\text{in}}}$ is real–valued in the range $[0.0, 1.0]$, and larger values denote a higher confidence. Consequently, $conf_{\text{prox}_{\text{in}}}$ can be used as a *quality measure* (Cellier 1991).

A position value for the m–output associated with the testing data can be estimated using a weighted sum of the m–outputs of the five nearest neighbors:

$$pos_{\text{out}} = \sum_{j=1}^{5} w_{\text{rel}}^{j} \cdot pos^{j} \tag{5.6}$$

The distance between the estimated m–output and any one of its five nearest neighbors is:

$$dis^j_{\text{out}} = \|pos_{\text{out}} - pos^j\| \tag{5.7}$$

The average distance used to determine the output confidence measure is computed as a weighted sum of the relative distances of the five nearest neighbors in the output space:

$$d_{\text{conf}_{\text{out}}} = \sum_{j=1}^{5} w^j_{\text{rel}} \cdot dis^j_{\text{out}} \tag{5.8}$$

The largest possible output distance value can be calculated as:

$$d_{\text{conf}_{\text{out}_{\text{max}}}} = n_{\text{out}} - 1 \tag{5.9}$$

where n_{out} is the number of classes of the m–output.

The confidence value related to the proximity of the five nearest neighbors in the output space can be defined as:

$$conf_{\text{prox}_{\text{out}}} = 1.0 - \frac{d_{\text{conf}_{\text{out}}}}{d_{\text{conf}_{\text{out}_{\text{max}}}}} \tag{5.10}$$

where $conf_{\text{prox}_{\text{out}}}$ is real–valued in the range [0.0, 1.0], and larger values denote a higher confidence. Consequently, $conf_{\text{prox}_{\text{out}}}$ can also be used as a quality measure.

Finally, the overall confidence is evaluated as the product of the individual confidence measures in the input and output spaces:

$$conf_{\text{prox}} = conf_{\text{prox}_{\text{in}}} \cdot conf_{\text{prox}_{\text{out}}} \tag{5.11}$$

5.4 The Similarity Measure

Measures of confidence can also be defined without the explicit use of a distance function. The input distance function is a scalar function over a vector space. This function throws potentially useful information about the position vectors away. Similarity measures avoid this problem by defining a similarity function between the position vectors themselves.

The similarity measure proposed in this chapter is a generalization of the classical set–theoretic equality functions. The generalization relies on the definitions of *cardinality* and *difference* in fuzzy set theory. The similarity measure presented in this section is based on intersection, union, and cardinality. It was originally proposed (in an entirely different context) by (Dubois and Pradé 1980).

$$S_1(A, B) = \frac{\|A \cap B\|}{\|A \cup B\|} \tag{5.12}$$

Clearly, when $A = B$, then $S_1(A, B) = 1.0$, and when A and B are totally disjoint, then $S_1(A, B) = 0.0$, i.e., the similarity function $S_1(A, B)$ can serve as a quality measure.

Up to this point, the position values p_i have been normalized in the range $[1.0, n_i]$, where n_i is the number of classes associated with the variable i_i. For the purpose of defining a similarity measure, it is more appropriate to re–normalize the position values into the range $[0.0, 1.0]$:

$$q_i = \frac{p_i - 1}{n_i - 1} \tag{5.13}$$

The q_i variables assume values in the range $[0.0, 1.0]$. Similarly, a re–normalized position value for the i^{th} m–input of the j^{th} nearest neighbor in the experience data base can be computed as:

$$q_i^j = \frac{p_i^j - 1}{n_i - 1} \tag{5.14}$$

The similarity of the i^{th} m–input of the j^{th} nearest neighbor to the testing m–input based on intersection can then be defined as follows:

$$sim_i^j = \frac{\min(q_i, q_i^j)}{\max(q_i, q_i^j)} \tag{5.15}$$

The overall similarity of the j^{th} neighbor is defined as the average similarity of all its m–inputs in the input space:

$$sim_{\text{in}}^j = \frac{1}{n} \sum_{i=1}^{n} sim_i^j \tag{5.16}$$

The position value of the m–output of the j^{th} neighbor can be re–normalized as follows:

$$q^j = \frac{p^j - 1}{n_{\text{out}} - 1} \tag{5.17}$$

where n_{out} denotes the number of classes associated with the output variable. A normalized position value for the testing m–output can be estimated using a weighted sum of the re–normalized position values of the m–outputs of the five nearest neighbors:

$$p_{\text{out}} = \sum_{j=1}^{5} w_{\text{rel}}^{j} \cdot p^{j} \tag{5.18}$$

Notice that Eq.(5.18) indirectly introduces the concept of distance again, since the relative weights, w_{rel}^{j}, are depending on d^{j}.

The similarity of the j^{th} neighbor to the estimated testing m–output based on intersection can be defined as follows:

$$sim_{\text{out}}^{j} = \frac{\min(q_{\text{out}}, q^{j})}{\max(q_{\text{out}}, q^{j})} \tag{5.19}$$

A confidence value based on similarity measures can thus be defined in the following fashion:

$$conf_{\text{sim}} = \sum_{j=1}^{5} w_{\text{rel}}^{j} \cdot sim_{\text{in}}^{j} \cdot sim_{\text{out}}^{j} \tag{5.20}$$

Also $conf_{\text{sim}}$ is a quality measure, i.e., a real–valued quantity in the range $[0.0, 1.0]$, where values close to 1.0 denote a reliable forecast.

5.5 Applications

In this section, two separate applications are discussed. Both confidence measures are computed in parallel, and compared to each other to evaluate their effectiveness at predicting forecasting errors.

Figure 5.4 shows Series L that had been introduced in Chapter 3, superposing in the top portion of the figure the actual measurement data (solid line) with the forecast (dashed line). Underneath, the two confidence measures are plotted.

The results obtained are rather interesting. It can be observed that both confidence measures produce similar results, whereby the confidence values obtained by the similarity measure are, on average, lower than those obtained by the proximity measure. The similarity measure is somewhat more sensitive, and therefore more reliable.

In Chapter 3, the L Series had been classified as *natural* rather than *synthetic*, because the data are indeed taken from a laser experiment (Weigend and Gershenfeld 1994). Yet, the data can be explained easily by the well–known Lorenz equations:

$$\frac{dX}{dt} = \sigma \cdot Y - \sigma \cdot X$$

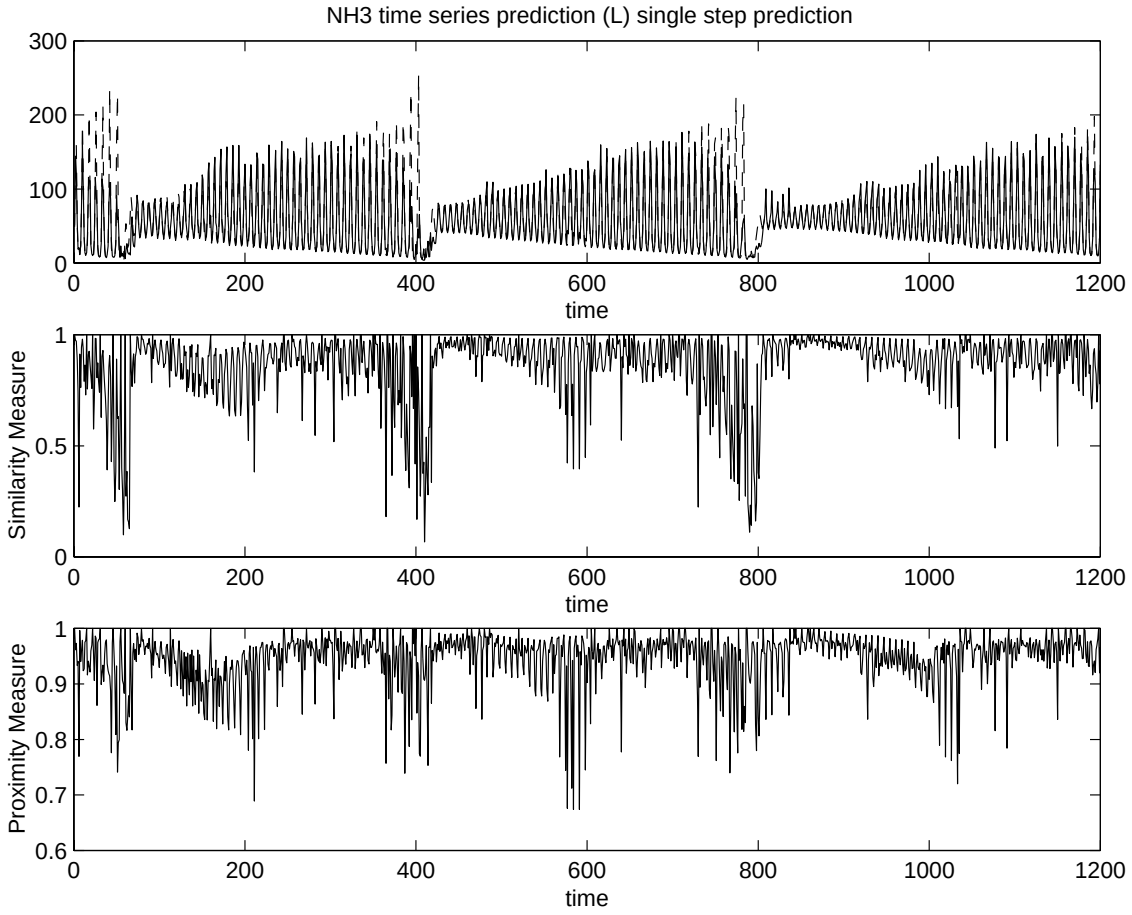

Figure 5.4: FIR confidence measures for Series L.

$$\begin{aligned} \frac{dY}{dt} &= -X \cdot Z + r \cdot X - Y \\ \frac{dZ}{dt} &= X \cdot Y - b \cdot Z \\ I &= X^2 \end{aligned} \tag{5.21}$$

The variable plotted in figure 5.4 is the laser intensity, I.

Hopf bifurcation occurs as a function of the Rayleigh number, r, keeping the Prandtl number, σ, and the parameter $b = 4\pi^2/(\pi^2 + k^2)$, where k is a dimensionless (normalized) wave number, constant.

The confidence values are generally much lower in the vicinity of the peaks of the laser intensity. This is meaningful, since the sensitivity to parameter variations in the Lorenz equations is indeed largest in the vicinity of the peaks. FIR detects this sensitivity, and provides considerably lower

confidence values in the vicinity of these peaks. It could be thought that the lower confidence values are caused by data deprivation. Since there are less data points available recording the behavior around the peaks, less good neighbors are available in the input space. However, this is not the major problem. With 10,000 samples, there are enough data points to find good neighbors also in the vicinity of the peaks. The confidence values are indeed reduced due to dispersion among the outputs of the five nearest neighbors, which is a reflection of the aforementioned increased parameter sensitivity.

Because of the high parameter sensitivity around the peaks, it is impossible to predict the amplitude of the next peak, although the behavior in between peaks can be predicted fairly well. Due to the same parameter sensitivity, also the time of the next switch–over event cannot be predicted precisely. Yet, as the amplitudes of the peaks grow, the probability of a switch–over event increases. FIR detects this also. As time progresses, FIR has less and less confidence in its predictions of peaks. Only after the next switch–over event took place, FIR gains renewed confidence in its predictions.

Yet, the Lorenz equations are even more interesting than that. Although the precise time of the next switch–over event cannot be predicted, the *average time between switch–over events*, i.e., the *average switch–over frequency (ASF)*, is prefectly predictable. For a given set of parameters $\{r, \sigma, b\}$, the value of ASF is constant. Yet, also ASF depends on r. As r increases, ASF increases with it. The laser operates at a value of r that is just below a bifurcation point of ASF, i.e., as r is slightly increased, ASF doubles. FIR picks even this bifurcation up. In the center between two neighboring switch–over events, FIR becomes more insecure, because it senses the onset of a switch–over event that never takes place, because r is a little too small. Once the danger is over, FIR becomes more confident again in its predictions, until the next switch–over event approaches.

In order to quantitatively test the effectiveness of the two confidence measures, a *local prediction error* needs to be defined. This can be done in a similar fashion to the error definition provided in Chapter 3, but the formula must be adjusted, because now, a point–wise error is needed.

The following approach is proposed. First, the observed testing data (**meas**) and the predicted testing data (**pred**) are jointly normalized to a range of $[0.0, 1.0]$:

$$M = \max(\mathbf{meas}, \mathbf{pred}) \tag{5.22}$$

$$m = \min(\mathbf{meas}, \mathbf{pred}) \tag{5.23}$$

$$mn_i = \frac{meas_i - m}{M - m} \tag{5.24}$$

$$pn_i = \frac{pred_i - m}{M - m} \tag{5.25}$$

The index i represents an individual measurement point, i.e., a point in time, when both the real data and the prediction were recorded.

Next, the local (point–wise) *absolute error* between the two normalized trajectories is computed:

$$err_{\mathrm{abs}_i} = |mn_i - pn_i| \tag{5.26}$$

Due to the previous normalization, the so computed absolute error can serve also as a measure of the relative error. Then, the *dissimilarity error* between the two normalized trajectories is computed:

$$simty_i = \frac{\min(mn_i, pn_i)}{\max(mn_i, pn_i, \epsilon)} \tag{5.27}$$

$$err_{\mathrm{sim}_i} = 1 - simty_i \tag{5.28}$$

Finally, the overall error is determined as the mean of the two errors computed above:

$$err_i = \frac{err_{\mathrm{abs}_i} + err_{\mathrm{sim}_i}}{2} \tag{5.29}$$

The local prediction error can be compared with the two "confidence errors," defined as:

$$err_{\mathrm{conf}_{\mathrm{sim}}} = 1 - conf_{\mathrm{sim}} \tag{5.30}$$

$$err_{\mathrm{conf}_{\mathrm{prox}}} = 1 - conf_{\mathrm{prox}} \tag{5.31}$$

Figure 5.5 shows the true *prediction error* plotted together with the *similarity error* and the *proximity error*. A strong correlation between the true error and the estimated errors is visible by naked eye.

Cross–correlations between the true and estimated prediction errors can be obtained using Matlab's *xcov* function. The results are shown in Figure 5.6.

Both curves exhibit strong positive cross–correlations at the center. However, the numerical value of the largest cross–correlation is five times bigger for the similarity measure than for the proximity measure. This result confirms, as had been stipulated before, that the similarity measure does a better job than the proximity measure at estimating the prediction error.

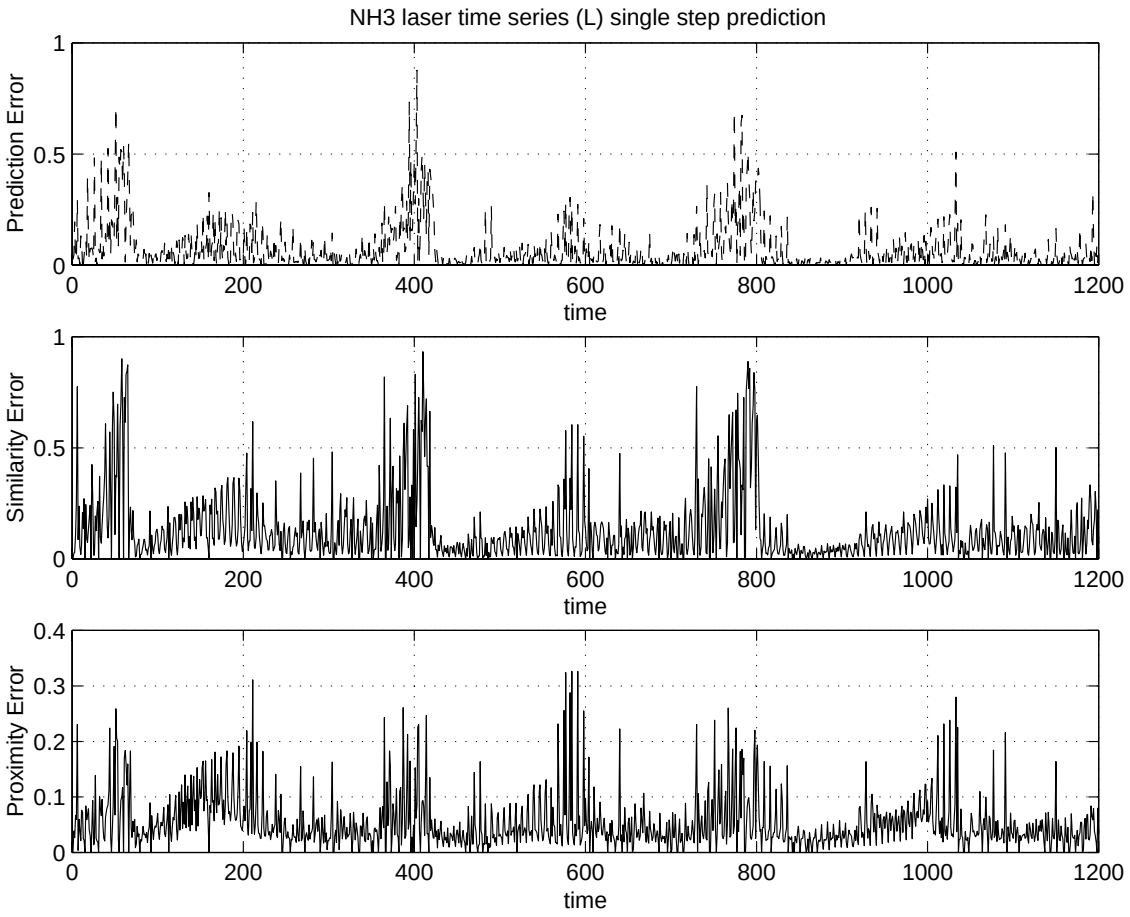

Figure 5.5: FIR true and estimated prediction errors for Series L.

Figure 5.7 shows once more Series B representing the water demand in the city of Barcelona. This time series has been introduced in Chapter 4. As before, the two confidence values are plotted underneath the measurement data that are superposed with the predictions obtained.

Contrary to Series L, the relationship between the prediction error and the confidence measures is not immediately evident. Due to the somewhat stochastic nature of this time series, the confidence is generally lower than in the case of Series L. The lower confidence values are primarily caused by a larger dispersion among the output values for similar inputs due to the stochastic nature of the data. In addition, the confidence is yet lower during weekends. This additional reduction in confidence is here an artifact of data deprivation. There are only about 70 weekends among the training data, and thus, there are less near neighbors in the input space for weekend days than

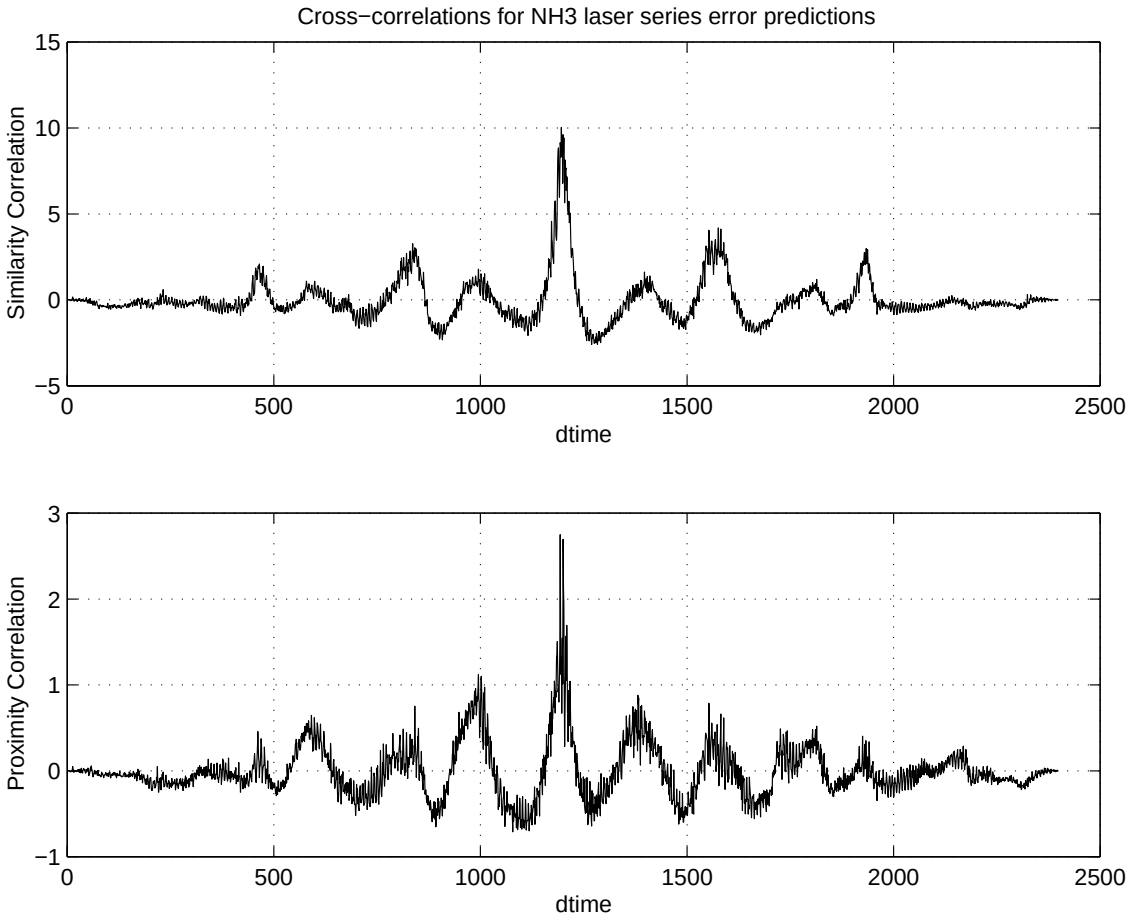

Figure 5.6: Cross–correlations between true and estimated prediction errors for Series L.

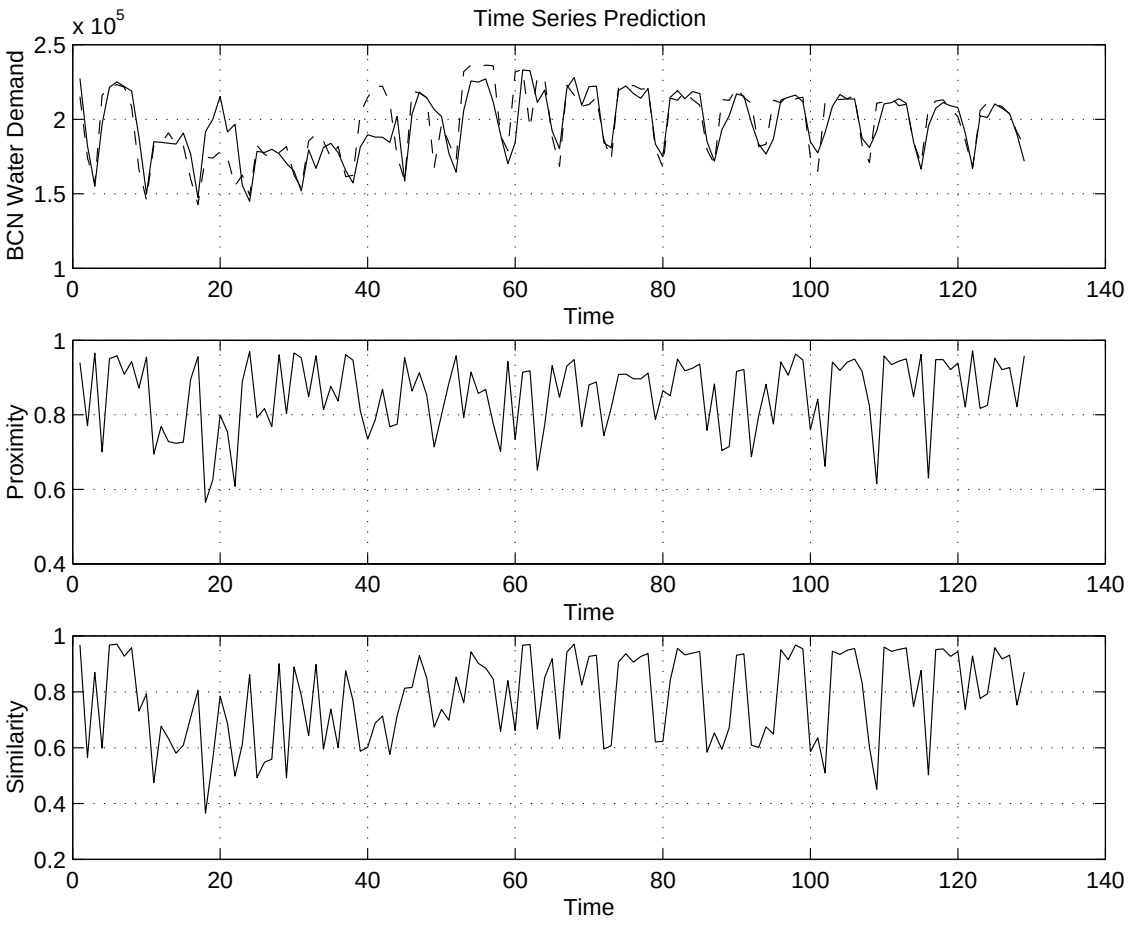

Figure 5.7: FIR confidence measures for Series B.

for week days. Hence the additional reduction in confidence is caused by a lack of good neighbors in the input space, rather than by dispersion among the neighbors in the output space.

As before, the true prediction errors are plotted together with the two estimated prediction errors, using the similarity and proximity confidence measures, respectively, are shown in Figure 5.8.

Although the relationship between the prediction error and the two confidence measures is not evident to the naked eye, it can be shown statistically. To this end, the cross–correlations between the true prediction error and the two estimated prediction errors are computed. They are shown in Figure 5.9.

As in the case of Series L, the two curves show a positive cross–correlation at the center. Also in this case, the peak of the cross–correlation

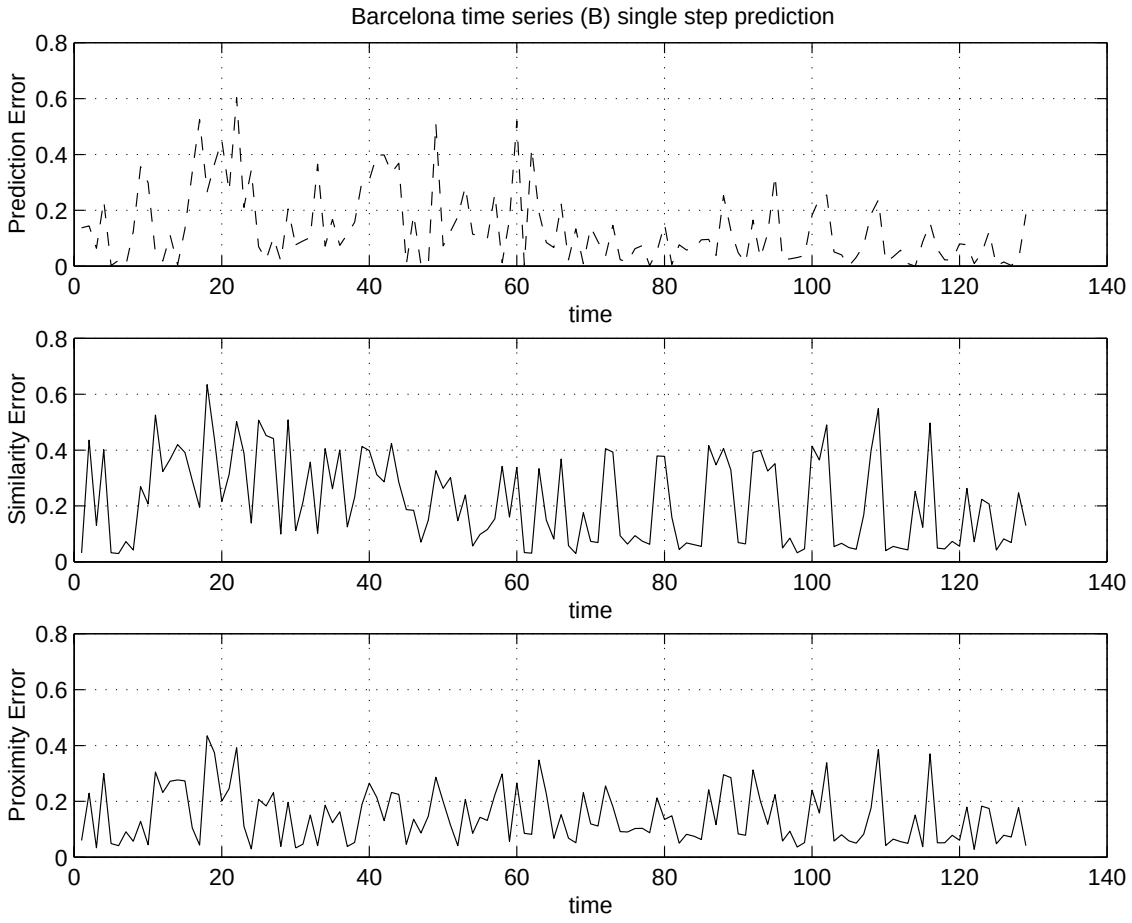

Figure 5.8: FIR true and estimated prediction errors for Series B.

for the similarity measure is higher than that for the proximity measure, demonstrating once more the superior performance of the similarity measure.

The two examples presented above are useful in analyzing the characteristics of the two proposed confidence measures, as well as the capabilities of FIR for prediction in two very different situations. In the deterministic case, represented here by Series L, the estimated prediction error is a true measure of the real prediction error, whereas in the stochastic case, here represented by Series B, the estimated prediction error is a measure of the true prediction error only in a statistical sense.

The value of confidence, using either of the two proposed confidence measures, is related to how deterministic the data base is, i.e., how close or disperse the outputs are for any one input pattern. When the process to be modeled is mostly deterministic, FIR will have a high level of confidence

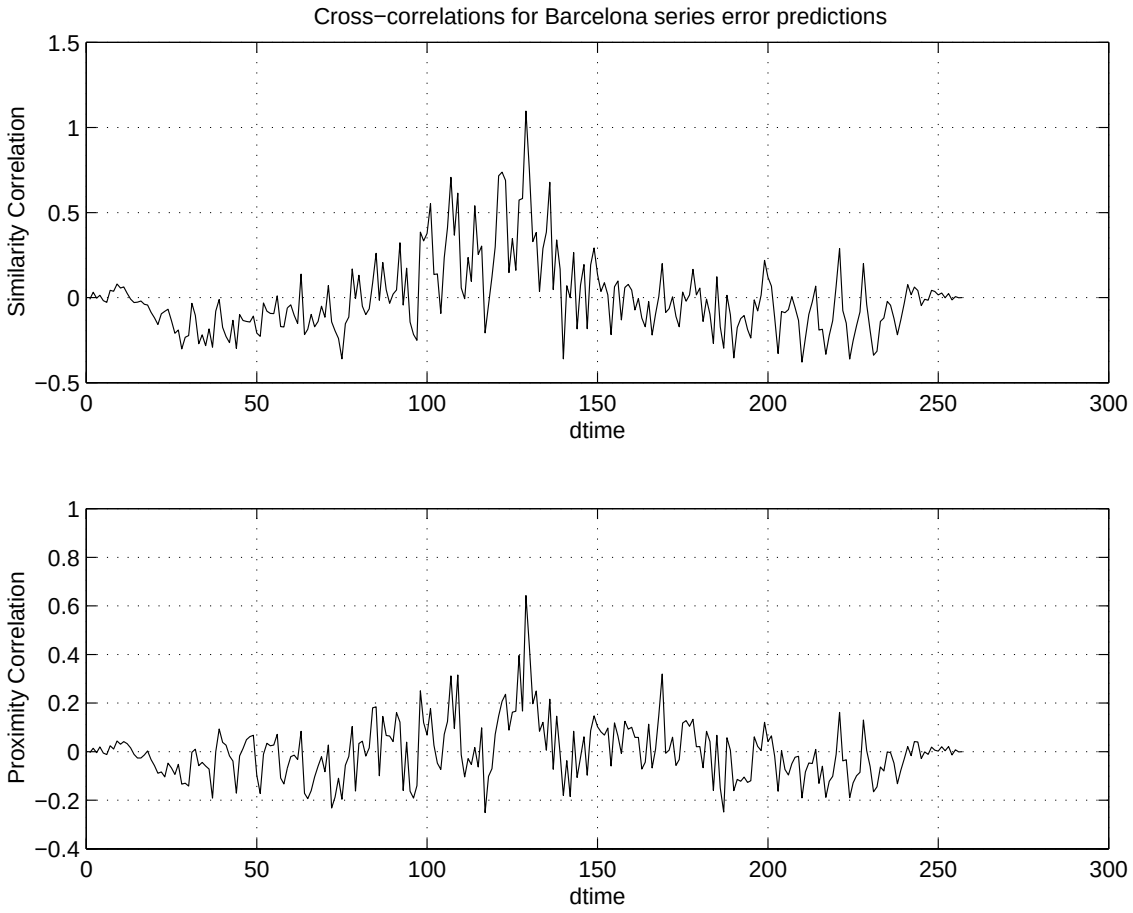

Figure 5.9: Cross–correlations between true and estimated prediction errors for Series B.

in the predictions it makes. A reduction of confidence, in this situation, is a reflection of data deprivation, i.e., FIR does not find enough close neighbors in the experience data base. More training data will cure the problem.

On the other hand, if the system to be modeled is stochastic in nature, FIR will exhibit a lower confidence in its predictions overall. The reason for the reduced confidence here is the dispersion of outputs among the five nearest neighbors. Additional training data will not be able to cure this problem.

5.6 Conclusions

When using fuzzy inductive reasoning models in prediction, it is very important to generate not only forecasts for the output variables, but also measures of the reliability of each forecast. Two measures of confidence in the reliability of FIR predictions have been proposed in this chapter, one being a *proximity measure*, the other being a *similarity measure*. After testing these measures on the largely deterministic time series L and on the somewhat stochastic time series B, a few conclusions can be drawn:

- The similarity measure is more sensitive to the prediction error than the proximity measure. This is reasonable, because the similarity measure preserves more information than the proximity measure about the qualitative difference between a new input state and its neighbors in the experience data base.

- Since the models derived by FIR are largely deterministic and autoregressive, in both the deterministic and the autoregressive stochastic processes, the proposed measures are useful tools to evaluate the likelihood of errors. More specifically, large proximity or similarity values indicate that a low prediction error is likely to occur.

- In time series corresponding to stochastic processes that are not entirely autoregressive, i.e., processes where the errors may be correlated, there is not necessarily a significant correlation between the prediction error and $(1.0 - conf_i)$. Therefore, the correlation between these two entities may, in general, be used as an indicator of how well the series in question may be fitted by an autoregressive or deterministic model.

A remark of a more philosophical nature is in place as well. The better the modeling methodology works, the less likely it is that a measure of the quality of the prediction can be made. If indeed the model were to exploit *all*

the information that is available in the measurement data, then the model of the prediction error would necessarily have to behave like uncorrelated white noise, because whatever can be said about the prediction error can, at least in theory, be exploited to improve the model. In practice, this is not a big problem. As long as the prediction error does not behave like white noise, the information obtained is useful to assess the quality of the prediction. On the other hand, once the prediction error starts to behave like white noise, the modeler can be assured that he or she has exploited every bit of knowledge available, and has come up with the best possible model already. Hence even in that case, the error analysis reveals something of value.

Chapter 6

Improving the Forecasting Capability of Fuzzy Inductive Reasoning by Means of Dynamic Mask Allocation

6.1 Introduction

In Chapter 5, a methodology was introduced that enables the FIR user to assess the quality of a prediction made. However, it was not attempted to come up with an estimate of the true prediction error directly. Instead, an indirect assessment was obtained in the form of a *confidence measure*.

It was mentioned that a direct attempt at estimating the prediction error must be futile as long as FIR does its job, because if it were possible to estimate the prediction error directly, then this estimate could be subtracted from the prediction, leading to an improved prediction. Such a naïve scheme was attempted already in Chapter 3 and shown not to work.

However, Chapter 5 also stipulated that *any* estimate of the prediction error, even an indirect one, can in principle, be used to improve the accuracy of a prediction made. After all, such an estimate *does* provide additional information about the prediction, an information that should be exploitable. This chapter presents one approach to exploit this information for improving the quality of predictions made.

The same approach can also be used to tackle yet another problem, namely that of dealing with *variable structure systems*. Some systems are *time–varying*. They change their behavioral patterns over time.

Many such systems operate in a number of different predefined *regimes*,

i.e., during some period of time, they exhibit similar behavioral patterns, and then, they suddenly switch from one operational mode to another. A car may serve as an example. It is in first gear during some period of time. Suddenly, the driver (or an automatic controller) decides to shift into second gear. The car now behaves differently from before.

Other systems are truly time–variant. They exhibit a continuous range of operational patterns. Here, an approach that classifies the behavioral patterns into discrete regimes is only an approximation of the true system complexity, yet, it may still be an effective way of enabling a person to make predictions of such a system.

In this chapter, it will be shown that FIR, together with the proposed methodology of dynamic mask allocation, can be used to deal with any and all of the above scenarios in a robust fashion.

6.2 The Concept of Dynamic Mask Allocation

The idea behind dynamic mask allocation is straightforward. In Chapter 3, it was shown that FIR, in its *qualitative modeling module*, proposes an *optimal mask*, i.e., a set of m–inputs that best characterize the output to be predicted.

Two separate quality measures were used to determine the optimal mask: the entropy reduction measure, H_r, that effectively measures the quality of information available, and an observation ratio measure, OR, that determines the quantity of information available. The mask quality was then determined as the product of the entropy reduction measure and the observation ratio measure:

$$Q = H_r \cdot OR \tag{6.1}$$

The optimal mask is the one that exhibits the largest Q value.

Yet, the selection of the optimal mask is by no means unique. There usually exist many masks of quite similar mask qualities (with similar Q values). Any of these masks can be used to make decent predictions. In fact, the *foptmask* routine of SAPS–II, the current implementation of FIR, returns not only the optimal mask, but the best mask of each complexity, in order to give the user a choice. Furthermore, a *mask evaluation report* can be requested that lists each mask that was tried together with its Q value.

It is quite reasonable to make multiple predictions in parallel using different masks of high quality. Until now, this was never done, because the user had no means to judge, which of the predictions obtained is the

best. Using either of the two confidence measures introduced in Chapter 5, this is now possible. Each of the predictions made using different masks comes with its own confidence estimate. It is then reasonable to accept, in each step, the one prediction that exhibits the largest confidence value.

Figure 6.1 demonstrates the algorithm.

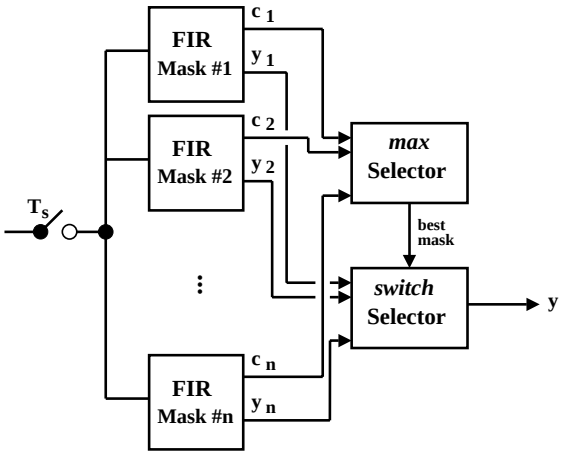

Figure 6.1: Dynamic mask allocation.

The switch at the left symbolizes the process of sampling, i.e., the passing of time. At each time step, n different FIR models (different masks) are used to make predictions in parallel. The variable y_i represents the predicted output using mask m_i, and c_i represents the estimated confidence in the prediction made. The different confidence values are then passed on to a *max selector* that determined the index, i, of the currently best mask:

$$i = \text{index of } \{c_i\}, \quad c_i = max! \tag{6.2}$$

The predicted outputs are fed into a *switch selector* that also receives the index of the currently best mask from the max selector. The switch selector passes through the y_i associated with the selected c_i:

$$y = y_i \tag{6.3}$$

The mask allocation is dynamic, because in each step, a different mask may be selected.

6.3 DMAFIR and QDMAFIR

The algorithm explained in the previous section has been named *DMAFIR*, denoting *Dynamic Mask Allocation for FIR*. The algorithm does not take into account the relative quality of the selected mask.

It might make sense to punish the use of masks of lower quality. To this end, a new quality measure is introduced:

$$Q_{\text{rel}_i} = \frac{Q_i}{Q_{\text{opt}}} \tag{6.4}$$

where Q_i is the mask quality of the selected mask, m_i, and Q_{opt} is the mask quality of the optimal mask. Clearly, Q_{rel_i} qualifies as a quality measure, since the value of Q_{rel_i} is in the range $[0.0, 1.0]$ with a larger value denoting the selection of a higher–quality mask. Q_{rel_i} is a *static mask quality* measure, as neither Q_i nor Q_{opt} change their values over time for any given mask, m_i.

Using this quality measure, the *dynamic mask quality* can be defined as:

$$Q_{\text{dyn}}(t) = Q_{\text{rel}_i}(t) \cdot conf_{\text{sim}}(t) \tag{6.5}$$

Here, $Q_{\text{rel}_i}(t)$ is indeed a function of time, because during each step, a different mask, m_i, may be chosen.

The so modified algorithm has been named *QDMAFIR*, denoting *Quality–adjusted Dynamic Mask Allocation for FIR*.

In the sequel, the two algorithms, DMAFIR and QDMAFIR, shall be applied to the water demand of the city of Barcelona to check whether dynamic mask allocation might help in obtaining better predictions. The rationale behind this experiment is that the water demand is quite different during weekends than during work days. Thus, if a mask is offered to FIR that makes better predictions for holidays, and another mask is provided that makes better predictions for working days, then FIR might automatically and dynamically choose the best mask in each case, offering overall better predictions than either of the individual masks might be able to generate.

6.4 Dynamic Mask Allocation Applied to Series B

Unfortunately, there are not enough data points available to train a model that predicts particularly well during weekends. Hence it was decided to offer to DMAFIR and QDMAFIR the top masks of complexities 2, up to 8, as proposed by the *mhis* matrix of the *foptmask* routine of SAPS–II. These masks, together with their qualities, are listed in Table 6.1.

Table 6.1: Suboptimal Masks and Their Qualities for Barcelona Time Series

Mask	Quality of the Mask
$y = \tilde{f}(y(t-\delta t), y(t-7\delta t), y(t-14\delta t))$	0.4539
$y = \tilde{f}(y(t-\delta t), y(t-3\delta t), y(t-7\delta t), y(t-12\delta t))$	0.3997
$y = \tilde{f}(y(t-\delta t), y(t-7\delta t))$	0.3879
$y = \tilde{f}(y(t-7\delta t))$	0.2993
$y = \tilde{f}(y(t-\delta t), y(t-3\delta t), y(t-5\delta t), y(t-11\delta t), y(t-14\delta t))$	0.2280
$y = \tilde{f}(y(t-\delta t), y(t-3\delta t), y(t-5\delta t), y(t-7\delta t), y(t-11\delta t)), y(t-14\delta t))$	0.0988
$y = \tilde{f}(y(t-\delta t), y(t-3\delta t), y(t-5\delta t), y(t-7\delta t), y(t-11\delta t)), y(t-13\delta t), y(t-14\delta t))$	0.0374

The best mask is a mask of complexity 4. It uses the values one day back, one week back, and two weeks back for its prediction. This is reasonable. The second best mask is a mask of complexity 5. The masks of yet higher complexity offer a considerably lower quality, because the amount of available data does not justify their use.

The mask quality is a compromise measure between two competing components. The *entropy reduction* measure assesses the uncertainty associated with a prediction, i.e., it is a measure of the *quality of information* available. The *observation ratio* measure judges the quality of neighbors, i.e., it is a measure of the *quantity of information* available.

Because of the lack of available training data, FIR cannot justify to always use a mask of high complexity. However, if at any point in time, there happen to be good neighbors available, then a mask of high complexity may offer a higher local quality, because it is associated with less uncertainty.

Both DMAFIR and QDMAFIR allow to exploit this. At any point in time, FIR will look for the proximity (or similarity) of its nearest neighbors, and it will pick the mask of highest complexity that offers neighbors that are sufficiently close.

Figure 6.2 compares the prediction errors of FIR when using only the optimal mask with that of FIR using the DMAFIR algorithm, once with the similarity measure, and once with the proximity measure.

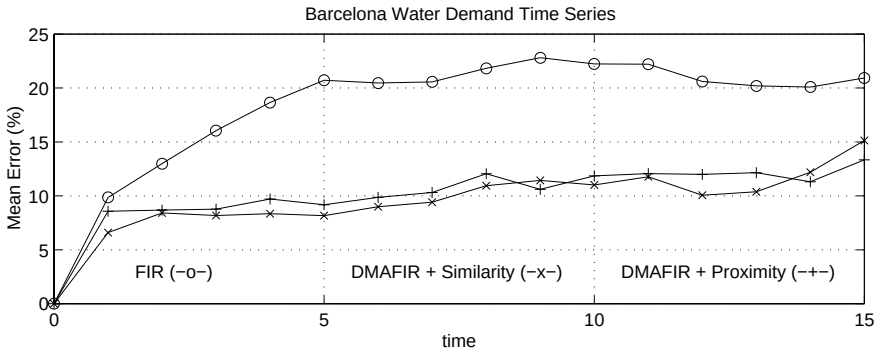

Figure 6.2: Comparison of FIR and DMAFIR for Barcelona time series.

There is a dramatic reduction in prediction errors. The proximity and similarity measures offer similar error reductions, with the similarity measure being slightly better on average.

Figure 6.3 compares the prediction errors of FIR when using only the optimal mask with that of FIR using the QDMAFIR algorithm, once with the similarity measure, and once with the proximity measure.

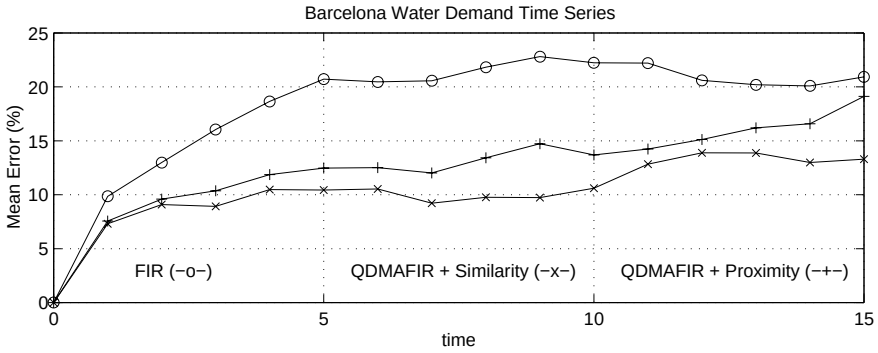

Figure 6.3: Comparison of FIR and QDMAFIR for Barcelona time series.

The results are quite similar to those found above. However in this case, the similarity measure offers a consistently larger error reduction than the proximity measure.

From now on, only the similarity measure will be used, because it was shown experimentally to be the better overall measure of the two.

Figure 6.4 compares the prediction errors of FIR when using only the optimal mask with that of FIR using the DMAFIR and QDMAFIR algorithms together with the similarity measure.

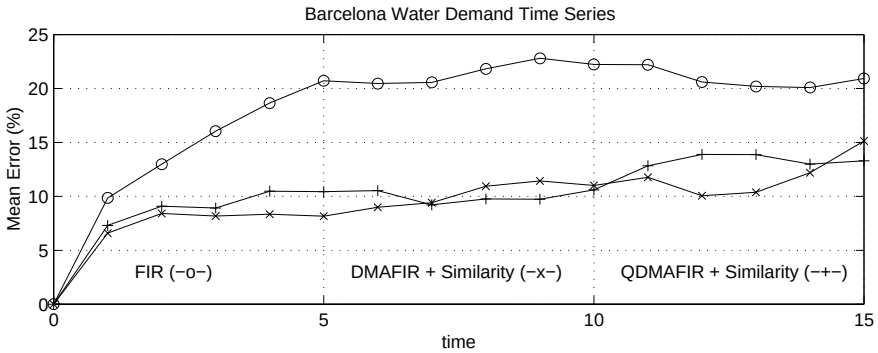

Figure 6.4: Comparison of FIR, DMAFIR, and QDMAFIR for Barcelona time series.

It turns out that the DMAFIR algorithm offers better results than the QDMAFIR algorithm. This is understandable. QDMAFIR gives a preference to masks that are close to the optimal mask in complexity. This hampers the ability of the algorithm to pick the mask of highest complexity that locally offers good neighbors.

The reader might have noticed by now that the results shown for FIR with the optimal mask are different from those shown in Chapter 4. The

reason is that, in Chapter 4, *the best results* that could be obtained for the Barcelona water series using a FIR qualitative simulation were compared with the other techniques. The method by which these results were obtained made use of dynamic mask allocation. In that method, QDMAFIR was applied to masks of low complexity, whereas DMAFIR was used for masks of higher complexity.

The comparisons made in Chapter 4 were generally fair, because most of the other estimation techniques do not offer self–assessment capabilities that would enable us to improve their predictions using a trick similar to that used in DMAFIR and/or QDMAFIR. This is true for all methods but one: the *FIR qualitative prediction*. In Figure 4.13, a comparison was made between the FIR qualitative simulation *with* dynamic mask allocation, and the FIR qualitative prediction *without* dynamic mask allocation. This comparison was unfair, because it would be perfectly feasible to apply dynamic mask allocation also to the FIR qualitative prediction.

In Figure 6.5, Figure 4.13 is repeated once more, this time comparing the FIR qualitative simulation *without* dynamic mask allocation with the FIR qualitative prediction *without* dynamic mask allocation.

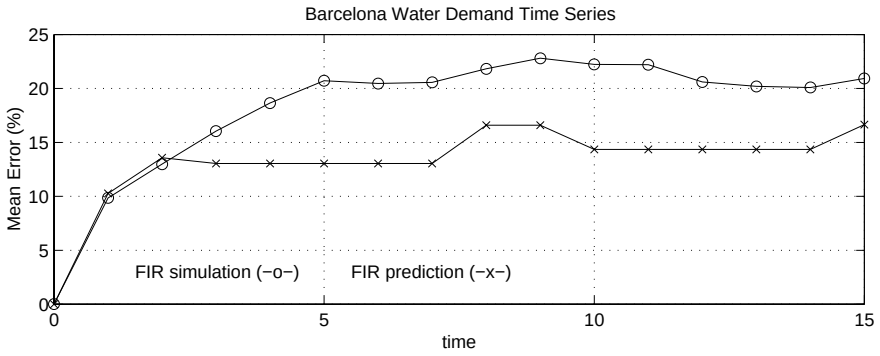

Figure 6.5: Comparison of FIR qualitative simulation and prediction without dynamic mask allocation for Barcelona time series.

The errors for the one–day prediction are identical, because the algorithms are the same for this case. However already in the case of a two–day prediction, the insecurity associated with the first–day prediction contaminates the available data so much that the FIR qualitative prediction algorithm outperforms the FIR qualitative simulation, in spite of the longer time horizon used by this estimator, and this remains true for all multiple–day predictions. Hence it is reasonable to suspect that an algorithm based on FIR qualitative prediction *with* dynamic mask allocation using DMAFIR together with the similarity confidence measure might beat even the best

results obtained so far.

Figure 6.6 compares the FIR qualitative simulation *with* dynamic mask allocation with the FIR qualitative prediction *with* dynamic mask allocation. Like in Chapter 4, the mask depth was enhanced from 15 to 22 for this experiment, so that the optimal mask algorithm would have m–inputs to pick even in the case of a 15–day prediction. Simultaneously, the maximum mask complexity was reduced from eight to seven, so that the most complex mask would just exhaust the available m–inputs for a 15–day prediction. Since the columns of the prediction matrix, Matrix 3.29, are independent of each other in the case of a qualitative prediction, they can be computed in parallel, which speeds up the execution time by a factor of 15.

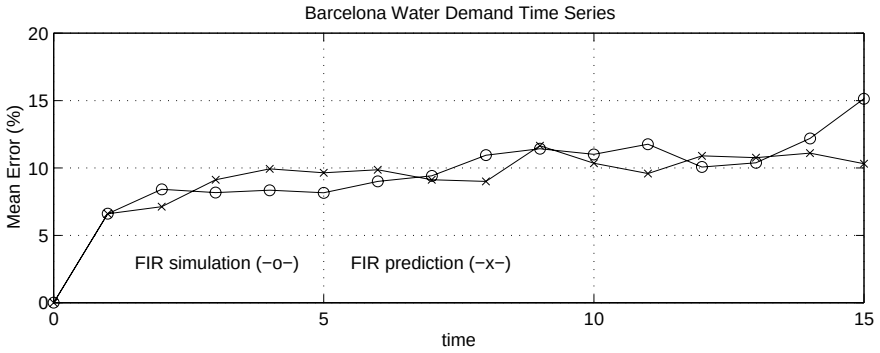

Figure 6.6: Comparison of FIR qualitative simulation and prediction with dynamic mask allocation for Barcelona time series.

As expected, the qualitative prediction is indeed somewhat better, on average, than the qualitative simulation. After two or three days of prediction, i.e., once the direct auto–correlation has died out, the error of the qualitative prediction, due to the quasi–stationary nature of Series B, hovers around 10%. In contrast, the error of the qualitative simulation suffers from data contamination, and therefore keeps growing. Because most of the models in use depend on the data points $(t-1)$ and $(t-7)$, the error grows in a staircase fashion between days $t+1$ and $t+2$, i.e., when the data point $(t-1)$ becomes contaminated, and then again between days $t+7$ and $t+8$, when also the data point $(t-7)$ becomes contaminated. The next step in the staircase occurs after two weeks, etc.

The distance between the qualitative simulation and prediction results is not as large in Figure 6.6 as it was in Figure 6.5. On the one hand, the dynamic mask allocation helps somewhat with desensitizing the predictions with respect to the data contamination problem, because FIR always picks the mask that offers the highest confidence. Since contaminated data lead

to a larger dispersion among the neighbors in the output space (reflecting their poorer information quality), the confidence measure is conscious of data contamination. On the other hand, the predictions obtained, due to the higher sophistication of the DMAFIR algorithm, is already much closer to the *theoretical limit of predictability*[1], i.e., any further improvement will invariably be quite modest.

What are the lessons that can be learned from this exercise?

1. The improvement of the forecasting quality obtainable by using a dynamic mask allocation algorithm is quite remarkable. Hence the fact that FIR offers a self–assessment capability is pivotal to its success in making predictions about the future behavior of time series. Prediction methods that do not offer a self–assessment capability, which are essentially all of FIR's contenders, are therefore severely disadvantaged.

2. Data contamination poses a serious threat to successful prediction of time series by any simulation approach. Consistently, the prediction methods beat the simulation methods, because they do not suffer from data contamination.

Can dynamic mask allocation save the day also in the case of Series R, i.e., the series describing the water demand of the city of Rotterdam? The author did apply both DMAFIR and QDMAFIR also to that time series. The reader will be spared a discussion of these experiments, as the attempts were futile. The results were as bad as those obtained in Chapter 4. There simply is no information contained in this time series that can be exploited beyond the direct auto–correlation between the water consumption of neighboring days, an auto–correlation that the trivial daily predictor exploits in an optimal fashion.

6.5 Predicting Time Series that Operate in Multiple Regimes: Series V

In this section, it will be demonstrated that the DMAFIR algorithm can be used to predict time series that operate in multiple regimes, i.e., where the behavioral patterns change between time segments. To this end, a new time series is introduced: Series V – the Van–der–Pol oscillator series.

[1] More information on the theoretical limit of predictability can be found in Chapter 8 of this dissertation.

The Van–der–Pol oscillator is described by the following second–order differential equation:

$$\ddot{x} - \mu \cdot (1 - x^2) \cdot \dot{x} + x = 0 \tag{6.6}$$

By choosing the outputs of the two integrators as two state variables:

$$\begin{aligned} \xi_1 &= x \\ \xi_2 &= \dot{x} \end{aligned} \tag{6.7}$$

the following state–space model is obtained:

$$\begin{aligned} \dot{\xi_1} &= \xi_2 \\ \dot{\xi_2} &= \mu \cdot (1 - \xi_1^2) \cdot \xi_2 - \xi_1 \\ y &= \xi_2 \end{aligned} \tag{6.8}$$

The ξ_2 variable is used as output of the time series.

Table 6.2 shows the characterization of Series V.

Table 6.2: Classification of Time Series V

natural		synthetic	V
stationary	V	non–stationary	
time invariant	V	time varying	
low dimensional	V	stochastic	
clean	V	noisy	
short		long	V
dormant		active	V
documented	V	blind	
linear		non–linear	V
scalar	V	vector	
single recording	V	multiple recordings	
continuous	V	discrete	

Series V is a synthetic time series, generated by a simulation model. Therefore, the data set can be made as long as needed. The Van–der–Pol oscillator is characterized by a stable limit cycle, i.e., already after the transitory period that is caused by the initial conditions imposed on the model has died down, a single limit cycle (one period of the oscillation) will

suffice to characterize the time series completely. The series is thus as active as it can ever be.

The behavioral patterns of Series V depend on the choice of the parameter μ. A time series operating in multiple regimes can be created by toggling between different values of μ in the course of the simulation.

To start the experiment, three different models were identified using three different values of μ, namely $\mu = 1.5$, $\mu = 2.5$, and $\mu = 3.5$. The first 80 data points of each time series were thrown away, as they represent the transitory period. The next 800 data points were used to learn the behavior of each series, and the subsequent 200 data points were used as testing data. With a sampling rate of 0.05, 200 data points correspond roughly to one oscillation period, i.e., four limit cycles were used for training the model, and one limit cycle was used for testing. The mask depth was chosen to be 50. All variables were classified into five classes with the landmarks -7.0, -0.5, -0.25, $+0.25$, $+0.5$, and $+7.0$. The same landmarks were used for all three time series, such that the results of the predictions can be more easily compared with each other.

The models obtained in this way are shown in Table 6.3.

Table 6.3: Optimal Masks and their Qualities for Series V

Regime	Optimal Mask	Quality of the Mask
$\mu = 1.5$	$y = \tilde{f}(y(t-\delta t), y(t-47\delta t))$	0.9342
$\mu = 2.5$	$y = \tilde{f}(y(t-\delta t))$	0.9085
$\mu = 3.5$	$y = \tilde{f}(y(t-\delta t))$	0.9146

The mask qualities are very high because of the strictly deterministic nature of Series V. The optimal masks for $\mu = 2.5$ and $\mu = 3.5$ are identical, yet the input/output behaviors will be different because of the different training data used by the two models.

Figure 6.7 compares the true time series with their predictions for each of the three models.

The top graph in Figure 6.7 compares the true Van–der–Pol cycle for $\mu = 1.5$ with the FIR predictions obtained using the model obtained for the same series. The graph below compares the Van–der–Pol data for $\mu = 2.5$ with the FIR predictions obtained using the corresponding FIR model, etc.

Because of the completely deterministic nature of this time series, the predictions should be perfect. They are not perfect due to data deprivation. Since 800 data points were used for training, the experience data base

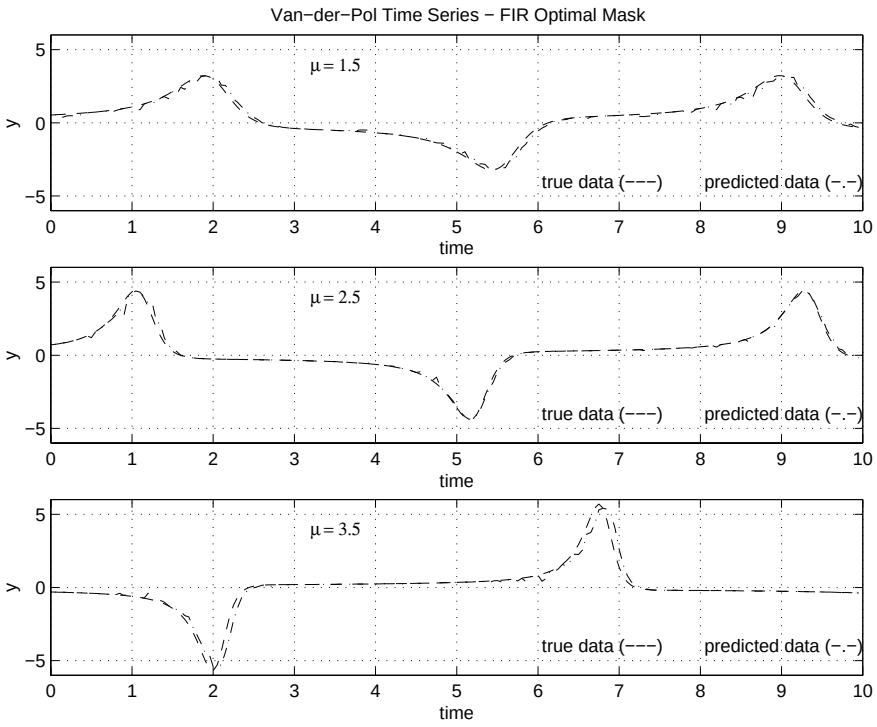

Figure 6.7: One–day predictions of the Van–der–Pol series using FIR without dynamic mask allocation.

contains only *four* cycles. Thus, when FIR, during the prediction, looks for *five* good neighbors, it only encounters *four* that are truly pertinent.

Figure 6.8 shows the predictions obtained when applying the model (optimal mask plus training data) obtained for the time series with $\mu = 1.5$ to the other two time series.

The model cannot predict the peaks of the time series with $\mu = 2.5$ and $\mu = 3.5$ correctly, because it has never seen such tall peaks. FIR can only predict behaviors that it has seen before.

Figure 6.9 shows the predictions obtained when applying the model (optimal mask plus training data) obtained for the time series with $\mu = 2.5$ to the other two time series.

The model predicts rather well the time series with $\mu = 1.5$, but has problems predicting the peaks of the time series with $\mu = 3.5$.

Figure 6.10 shows the predictions obtained when applying the model (optimal mask plus training data) obtained for the time series with $\mu = 3.5$ to the other two time series.

This model predicts all three time series rather well. Table 6.4 summarizes

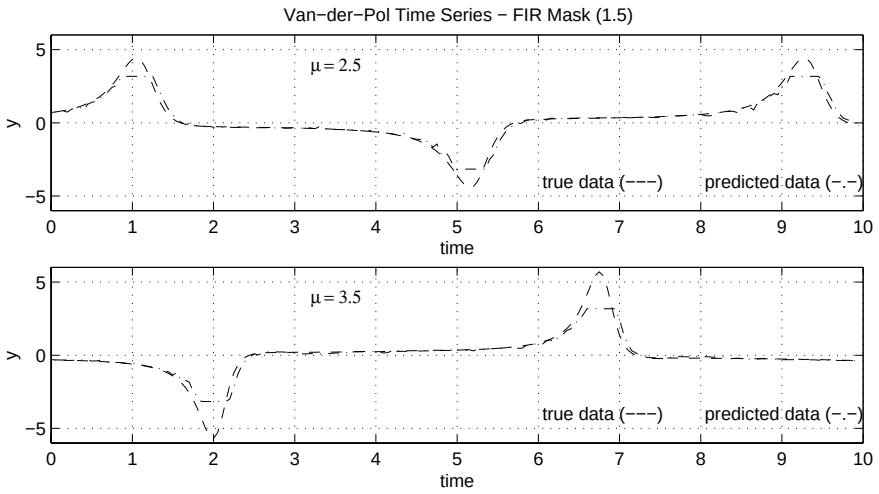

Figure 6.8: One–day predictions of the Van–der–Pol series using FIR with $\mu = 1.5$ model.

the errors obtained for all nine predictions.

Table 6.4: Prediction Errors for Series V

	Series ($\mu = 1.5$)	Series ($\mu = 2.5$)	Series ($\mu = 3.5$)
Model ($\mu = 1.5$)	2.5760	6.6957	11.5990
Model ($\mu = 2.5$)	3.5676	1.2256	3.6179
Model ($\mu = 3.5$)	4.1720	2.6618	8.7299

The model derived from the series with $\mu = 3.5$ predicts the other two series better than the series for which it was derived. This is due to the high–frequency components of this time series. The sharp gradients contribute to a considerably larger error. Thus, this result is understandable.

More suspicious is the result that the model obtained for $\mu = 2.5$ should be able to predict the time series with $\mu = 3.5$ better than the model obtained for $\mu = 3.5$, in spite of the fact that the former model cannot predict the peaks of this time series correctly. Indeed, if one compares visually the bottom curve of Figure 6.7 with the bottom curve of Figure 6.9, one would be inclined to believe that the quality of the former prediction is better than that of the latter, in spite of the larger numerical value of the error. Hence this result is a fluke of the particular error formula used.

What happened in this particular case is the following. The mean value

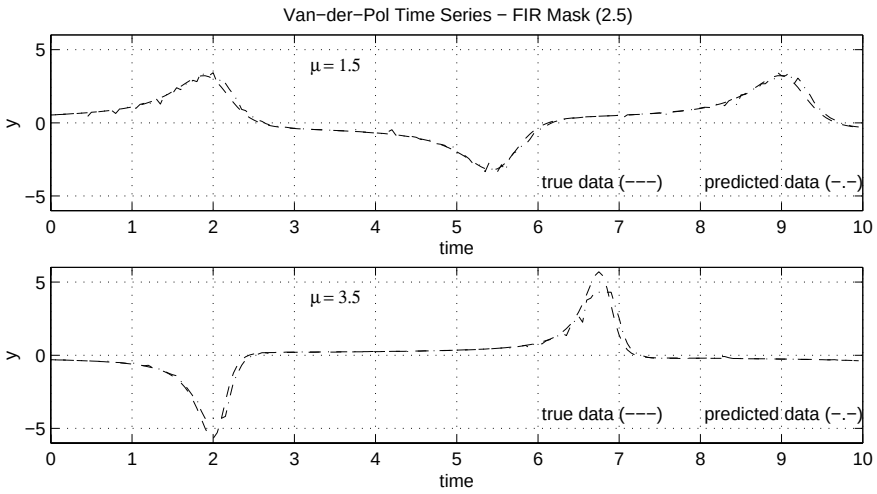

Figure 6.9: One–day predictions of the Van–der–Pol series using FIR with $\mu = 2.5$ model.

of the time series is very close to zero (-0.0164). The mean value of the prediction using the model obtained for $\mu = 3.5$ is -0.0232. Thus, the relative error between these two values that is used by the error formula proposed in Chapter 3:

$$err_{\text{mean}} = \frac{\|y_{1_{\text{mean}}} - y_{2_{\text{mean}}}\|}{\max(\|y_{1_{\text{mean}}}\|, \|y_{2_{\text{mean}}}\|, \varepsilon)} \tag{6.9}$$

obtains a very large numerical value of $err_{\text{mean}} = 29.44\%$. This number dominates the overall error formula by two orders of magnitude. Unfortunately, no error formula is ever perfect!

Thus, for this example, it makes sense to modify the error formula by eliminating the contribution of the mean value and dividing the total error by three rather than by four.

Table 6.5 shows the errors using the modified error formula.

Table 6.5: Prediction Errors for Series V Using Modified Error Formula

	Series ($\mu = 1.5$)	Series ($\mu = 2.5$)	Series ($\mu = 3.5$)
Model ($\mu = 1.5$)	2.6292	6.7597	10.3922
Model ($\mu = 2.5$)	2.9645	0.9747	4.6463
Model ($\mu = 3.5$)	4.2691	2.5744	1.8272

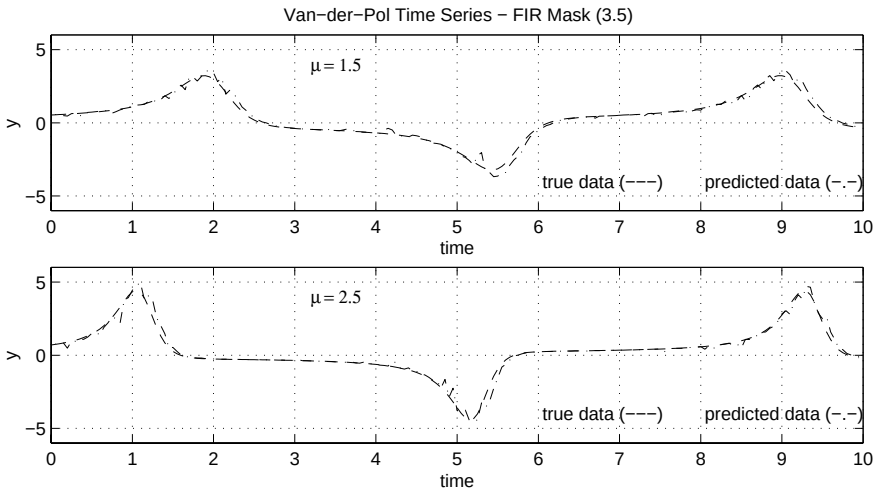

Figure 6.10: One–day predictions of the Van–der–Pol series using FIR with $\mu = 3.5$ model.

Now, the results are as they would have been expected. The values along the diagonal are smallest, and the values in the two remaining corners are largest. It also makes sense that the model obtained for $\mu = 3.5$ is more capable of predicting the series with $\mu = 1.5$ than the other way around.

Next, a time series shall be constructed, in which the variable μ assumes a value of 1.5 during one segment, followed by a value of 2.5 during the second time segment, followed by yet another time segment, in which $\mu = 3.5$. The multiple regimes series consists of 553 samples.

Figure 6.11 shows the results of predicting the multiple regimes series using the three models independently.

The model obtained for $\mu = 1.5$ cannot predict the higher peaks of the second and third time segment very well, therefore its error must be largest. The model obtained for $\mu = 3.5$ does a decent job at predicting all three segments. Thus, its error must be smallest.

Figure 6.12 shows the results of predicting the multiple regimes series using DMAFIR together with the similarity confidence measure. The three individual models (optimal masks plus training data sets) are offered to the DMAFIR algorithm to choose from.

The top plot shows the prediction obtained by DMAFIR. The bottom plot shows, which of the three models was chosen at any point in time. The value plotted is the μ–value of the chosen model. During the first time segment, consisting of the first 178 samples, the "average" μ–value is $\mu_{\text{avg}} = 1.9831$. During the second segment, the average μ–value is $\mu_{\text{avg}} = 2.4831$. Finally,

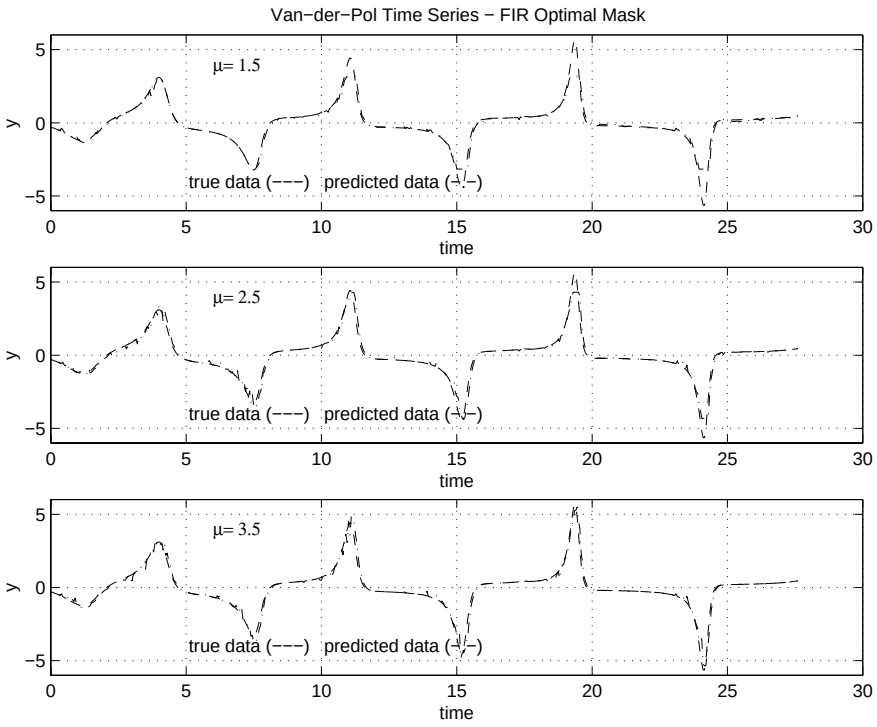

Figure 6.11: One–day predictions of the Van–der–Pol multiple regimes series.

during the third time segment, the average μ–value is $\mu_{\text{avg}} = 3.0871$. Thus, on average, FIR indeed picks more often than not the correct model.

Table 6.6 lists the prediction errors obtained for the different simulations using the modified error formula.

Table 6.6: Prediction Errors for Multiple Regimes Series V Using Modified Error Formula

	error
Model for $\mu = 1.5$	5.8759
Model for $\mu = 2.5$	2.2978
Model for $\mu = 3.5$	1.9317
DMAFIR	1.1195

As was to be expected, the model obtained for $\mu = 3.5$ shows the smallest of the individual errors. However, the error obtained using DMAFIR is still considerably smaller. This demonstrates that DMAFIR can indeed be

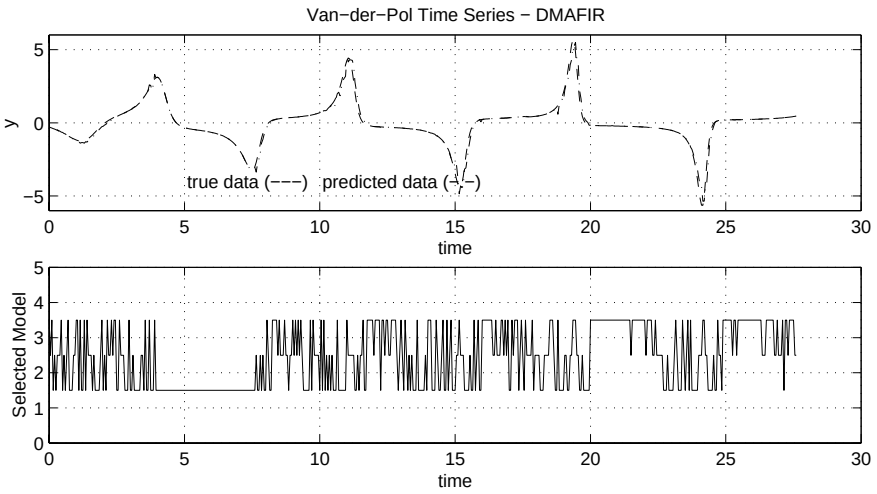

Figure 6.12: One–day predictions of the Van–der–Pol multiple regimes series using DMAFIR.

successfully applied to the problem of predicting time series that operate in multiple regimes.

6.6 Variable Structure System Prediction Using FIR With Dynamic Mask Allocation

In this section, it will be shown that the DMAFIR algorithm can be successfully employed for predicting time–varying systems. Whereas a system that operates in multiple regimes exhibits a fixed number of different behavioral patterns, a time–varying system exhibits an entire spectrum of different behavioral patterns.

To demonstrate DMAFIR's ability of dealing with time–varying systems, the Van–der–Pol oscillator is used once again. This time, a series was generated, in which μ changes its value constantly in the range $[1.0, 3.5]$. The time series contains 953 records sampled using a sampling interval of 0.05. The value of μ changes once per sample.

Figure 6.13 shows the results of predicting the time–varying series using the three models independently.

Each peak is of slightly different amplitude, i.e., the time–varying Van–der–Pol oscillator series is no longer completely deterministic. As expected, the model obtained for $\mu = 3.5$ works best, because it has no difficulty predicting the high–amplitude peaks. Also the model obtained for $\mu = 2.5$

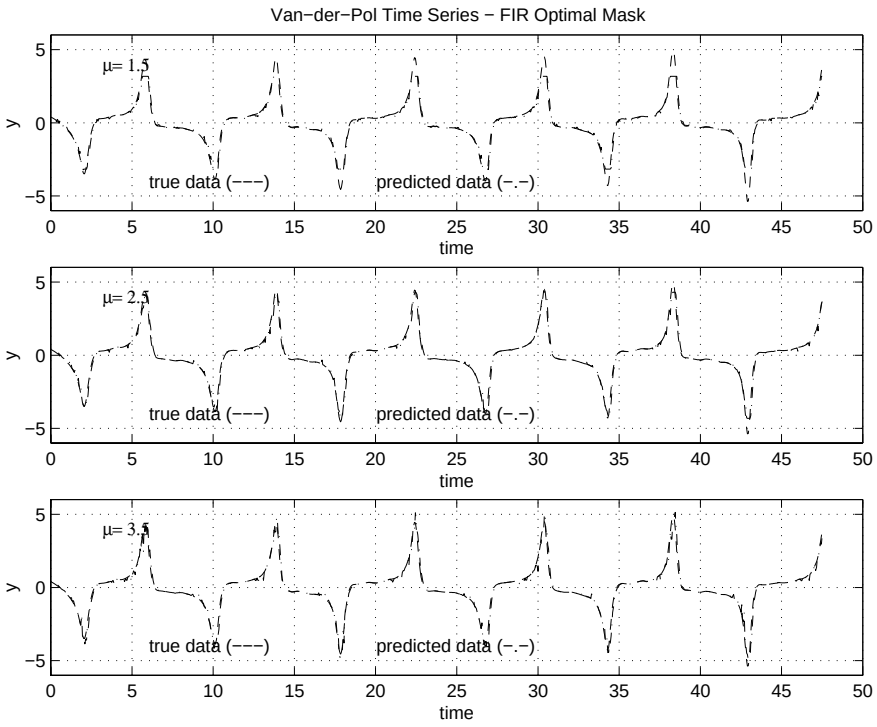

Figure 6.13: One–day predictions of the Van–der–Pol time–varying series.

works very well, because the system has low–pass characteristics. Although μ varies in the range $[1.0, 3.5]$, the extremely small and extremely large peaks characteristic of very small and very large μ values never show up in the simulation results. The model obtained for $\mu = 1.5$ is least suitable, because it cannot predict high–amplitude peaks that it has never seen during the training phase.

Of course, it would have been possible to make the methodology adaptive by augmenting the training data base with new input/output pairs as they become known during the testing period. Yet, it was decided to exclude adaptive schemes from the research presented in this dissertation, since this would have opened an entirely new dimension to the research. Questions would need to be answered relating to the monotonicity of the available knowledge, i.e., while a time–varying system changes its behavior, it may be appropriate, not only to *add* new knowledge as it becomes available, but also to *discard* previous knowledge that is in contradiction with new knowledge obtained. This leads into the area of *non–monotonic reasoning* (Sarjoughian 1995), an extensive research field in its own right that the author decided not to delve into as part of her dissertation.

Figure 6.14 shows the results of predicting the time–varying Van–der–Pol series using DMAFIR together with the similarity confidence measure. The three individual models (optimal masks plus training data sets) were offered to the DMAFIR algorithm to choose from.

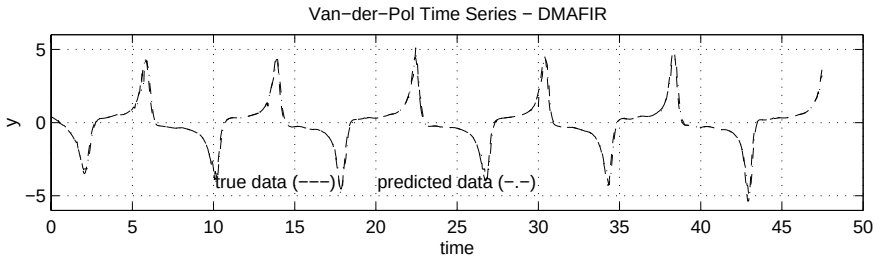

Figure 6.14: One–day predictions of the Van–der–Pol time–varying series using DMAFIR.

The prediction is close to perfect. As expected, DMAFIR makes the prediction more robust, and reduces the prediction error to a level that is below that obtainable by either of the individual models.

Table 6.7 lists the prediction errors obtained for the different simulations using the modified error formula.

Table 6.7: Prediction Errors for Time–Varying Series V Using Modified Error Formula

	error
Model for $\mu = 1.5$	5.7431
Model for $\mu = 2.5$	1.4864
Model for $\mu = 3.5$	1.8791
DMAFIR	1.2997

The experiment shows that DMAFIR is indeed capable of dealing with variable structure system predictions. Although such systems do not have a finite set of individual behavioral patterns, it is useful to discretize the spectrum of behavioral patterns, identify individual models for each of these patterns, and then let DMAFIR choose among them during the variable structure system prediction.

6.7 Conclusions

In this chapter, a methodology was introduced that allows to exploit the confidence measure of FIR, an indirect prediction error estimate, for improving the predictions made.

It was shown in Chapter 3, that a direct error estimate coupled with an error subtraction scheme does not work. The confidence measure was subsequently introduced in Chapter 5 as a means to indirectly assess the quality of a prediction made. The present chapter showed how this information can be exploited to improve the quality of predictions made.

It was shown that the self–assessment capability of FIR is pivotal to its capability of making high–quality predictions of time series. In the case of Series B, FIR fares better than ARIMA and/or ANN *only* if a dynamic mask allocation scheme is used to improve its forecasts. FIR applied in the traditional fashion relying on the optimal mask alone would not have won this race.

As a side product, it was shown that the data contamination problem that haunts any and all time series simulation schemes must indeed be taken seriously. Time series prediction schemes will generally fare better than their simulation cousins, because they are not haunted by data contamination.

Chapter 7

Predicting the Predictability Horizon

7.1 Introduction

In Chapter 5 of this thesis, the problem of estimating the forecasting error in time series predictions was discussed. It was shown that, especially in soft science simulation, it is important to estimate the error of a prediction together with the prediction itself, since it cannot be expected of the users that they would be able to assess the reliability of the simulation results. Scepticism must be instilled in the simulation software, rather than demanding it of its users.

The present chapter deals with a closely related topic. Since the simulation results cannot be expected to be totally accurate, errors are likely to accumulate during iterative predictions of future values of a time series. It is thus of much interest to the user of such a tool to be able to assess the quality of predictions made not only locally, but as a function of time, i.e., the user should be able to obtain a (generally decaying) function of *accumulated confidence* in progressive predictions. During the first step of a multi–step prediction, the predicted value depends entirely on measurement data, and is therefore more likely to be accurate than in subsequent steps, when the predictions depend on previously predicted data points that are by themselves associated with a degree of uncertainty already. The effects of error accumulation due to data contamination have been demonstrated in Chapters 4 and 6 of this thesis.

There exist many applications for such a technology. For example, model predictive control uses predictions of future values of measurement data to provide the controller with an early warning if the system is about to leave

the zone of safe operation. The earlier such a warning can be provided, the more time the controller has to prevent this situation from ever taking place.

Yet, there are two types of errors that can occur in such predictions:

1. The predictor foresees that the system will leave its operating zone, although in reality, this would not take place.

2. The predictor does not foresee any problems, although they do take place.

Both types of errors can degrade the achievable performance of the controller. The first error type will make the controller overly conservative, preventing it from making use of the full operating zone. The second error type may lead to either instability or plant shutdown.

Both error types are closely related to the horizon of predictability. As the accuracy of forecasts in a multi–step prediction decreases over time, the likelihood of committing either type of error grows. Hence assessing the likelihood of these errors to occur is synonymous with being able to assess the horizon of predictability of each measurement signal used in the predictive control scheme.

The chapter introduces measures for estimating the horizon of predictability. It then calculates the prediction errors made when forecasting three separate time series over multiple steps, and shows the strong positive correlation between the prediction error on the one hand and the estimated horizon of predictability on the other.

7.2 Accumulated Confidence Measures in Time–Series Prediction

As was shown in Chapter 5, the local prediction error can be indirectly estimated using either a *proximity* or a *similarity* measure. Both types of estimators lead to satisfactory results when used together with a FIR algorithm for time–series prediction, although the similarity measure is usually preferred, as it is slightly more sensitive than the proximity measure.

Both measures only account for uncertainty stemming from a single step of prediction, i.e., they assume that the data on which the prediction is based are totally accurate. They measure the *local* uncertainty associated with a single prediction, but not the *accumulated* uncertainty resulting from multiple predictions, whose premises are themselves uncertain already.

Either measure can easily be extended to become an estimator of accumulated confidence. The reader may remember that a FIR model of

a time–series predictor is characterized by a single–column optimal mask, e.g.:

$$\begin{array}{cc} {}_t\backslash^x & y \\ t-5\delta t & \begin{pmatrix} -1 \\ 0 \\ 0 \\ -2 \\ -3 \\ +1 \end{pmatrix} \\ t-4\delta t & \\ t-3\delta t & \\ t-2\delta t & \\ t-\delta t & \\ t & \end{array} \tag{7.1}$$

denoting the equation:

$$y(t) = \tilde{f}(y(t-5\delta t), y(t-2\delta t), y(t-\delta t)) \tag{7.2}$$

where $\tilde{f}$ denotes a function specified through a finite state machine, rather than being provided in the form of an analytical expression.

Negative mask elements denote mask inputs (m–inputs), whereas the $+1$ element, which will always show up in the last row, denotes the mask output (m–output).

For the above mask, it makes sense to define the accumulated confidence in the prediction of $y(t)$ as follows:

$$c_a(t) = c_l(t) \cdot \frac{1}{3} \cdot (c_a(t-5\delta t) + c_a(t-2\delta t) + c_a(t-\delta t)) \tag{7.3}$$

i.e., the accumulated confidence in the prediction of $y(t)$, called $c_a(t)$, is defined as the product of the local confidence in that prediction, $c_l(t)$, with the average accumulated confidence in the three m–inputs. It would have been equally acceptable to define the *joint accumulated confidence* of the m–inputs in other ways, such as:

$$c_{a_{\text{joint}}} = \min(c_a(t-5\delta t), c_a(t-2\delta t), c_a(t-\delta t)) \tag{7.4}$$

or:

$$c_{a_{\text{joint}}} = c_a(t-5\delta t) \cdot c_a(t-2\delta t) \cdot c_a(t-\delta t) \tag{7.5}$$

and then:

$$c_a(t) = c_l(t) \cdot c_{a_{\text{joint}}} \tag{7.6}$$

Either of these techniques will work, but the one adopted in this dissertation is the one proposed in Eq.(7.3).

Clearly both the local and accumulated confidence values of measured data points are 1.0, and therefore, the accumulated confidence of the first prediction step is always equal to the local confidence, computed using either the proximity or the similarity measure, but at later times, the accumulated confidence is always lower than the local confidence. The accumulated confidence is usually decaying over time, although it is not necessarily a monotonically decreasing function.

The multiplication of the local confidence of the m–output with the average accumulated confidence of the m–inputs is only correct, in a strict sense, if subsequent values of y can be assumed to be uncorrelated, which, of course, is never the case. However from a practical stand point, the measure works exceedingly well, as shall be demonstrated by means of three separate examples. The accumulated confidence was already informally introduced in Chapter 3 of this thesis (starting from Figure 3.13), however without going into any details as to how the accumulated confidence is actually computed.

Of course, the proposed approach to estimating the accumulated confidence in qualitative predictions is not limited to time series. For example, given a system with two inputs and three outputs characterized by the following optimal mask:

$$\begin{array}{c} {}_t\backslash^x \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} \begin{array}{ccccc} u_1 & u_2 & y_1 & y_2 & y_3 \end{array} \\ \begin{pmatrix} -1 & 0 & -2 & 0 & 0 \\ 0 & -3 & 0 & 0 & -4 \\ 0 & 0 & +1 & 0 & 0 \end{pmatrix} \end{array} \tag{7.7}$$

denoting that:

$$y_1(t) = \tilde{f}(u_1(t-2\delta t), y_1(t-2\delta t), u_2(t-\delta t), y_3(t-\delta t)) \tag{7.8}$$

would lead to the following expression of accumulated confidence:

$$c_a(y_1(t)) = c_l(y_1(t)) \cdot (0.5 + 0.25 \cdot c_a(y_1(t-2\delta t)) + 0.25 \cdot c_a(y_3(t-\delta t))) \tag{7.9}$$

Since the input variables are always measured and therefore assumed to be accurate, the accumulated confidence values associated with $u_1(t-2\delta t)$ and $u_2(t-\delta t)$ is always assumed to be 1.0. This time it was necessary to specify the names of the variables as arguments of the c_a and c_l functions, since multiple variables are contributing to the prediction.

7.3 Simulation Results

Three separate time series were used to investigate the effectiveness of the proposed accumulated confidence measures as indirect statistical estimators for the prediction error to be expected.

The first time series represents the water demand of the City of Barcelona (Series B), the second series represents the water demand of the City of Rotterdam (Series R), and the third time series represents the temperature of the City of Tucson (Series T). Series T is a time series that is newly introduced in this chapter.

7.3.1 Water Demand of the City of Barcelona: Series B

This time series has been introduced in Chapter 4, and was reused in Chapter 6. Figure 4.5 is repeated once more in Figure 7.1.

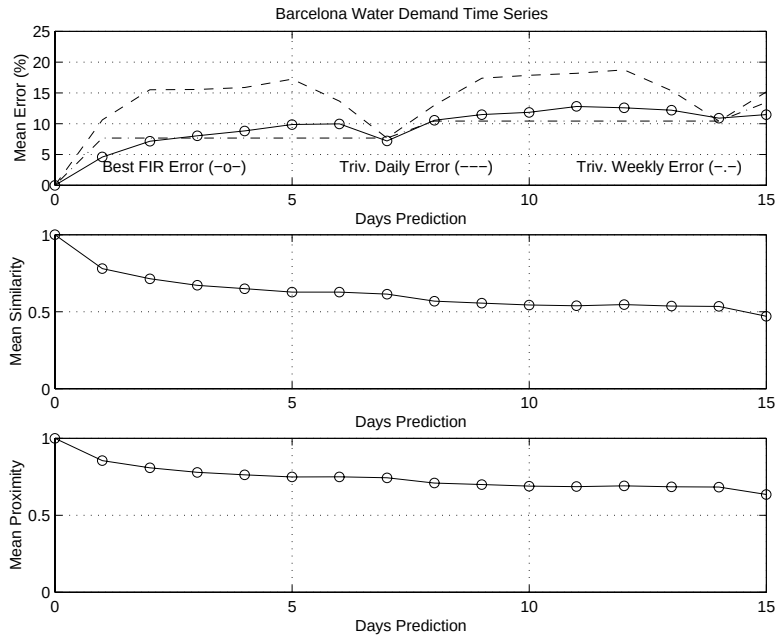

Figure 7.1: Barcelona water demand multiple–step simulation using FIR.

The lower portions of Figure 7.1 were introduced informally in Chapter 4 as an intuitive means to representing the confidence made in the prediction. However, no explanation was presented in Chapter 4, how these curves have been obtained.

The center curve shows the *average accumulated confidence* in the predictions made as a function of the number of days predicted, using a formula that corresponds to that of Eq.(7.3). Since dynamic mask allocation was used, the formula for the accumulated confidence changes every day in accordance with the mask being used on that day. On days when the optimal mask:

$$y(t) = \tilde{f}\left(y(t-\delta t), y(t-7\delta t), y(t-14\delta t)\right) \tag{7.10}$$

is being used, the accumulated confidence is computed using the formula:

$$c_a(t) = c_l(t) \cdot \frac{1}{3} \cdot \left(c_a(t-\delta t) + c_a(t-7\delta t) + c_a(t-14\delta t)\right) \tag{7.11}$$

whereas on days when the next best mask:

$$y(t) = \tilde{f}\left(y(t-\delta t), y(t-3\delta t), y(t-7\delta t), y(t-12\delta t)\right) \tag{7.12}$$

is being used, the accumulated confidence is computed using the formula:

$$c_a(t) = c_l(t) \cdot \frac{1}{4} \cdot \left(c_a(t-\delta t) + c_a(t-3\delta t) + c_a(t-7\delta t) + c_a(t-12\delta t)\right) \tag{7.13}$$

etc.

The results are written into a matrix of accumulated confidences that has the same structure as Matrix (3.29):

$$C_a = \begin{pmatrix}
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
1.0 & \mathrm{c_a(t-3\delta t, 1)} & \mathrm{c_a(t-2\delta t, 2)} & \mathrm{c_a(t-\delta t, 3)} & \mathrm{c_a(t, 4)} & \ldots \\
1.0 & \mathrm{c_a(t-2\delta t, 1)} & \mathrm{c_a(t-\delta t, 2)} & \mathrm{c_a(t, 3)} & \mathrm{c_a(t+\delta t, 4)} & \ldots \\
1.0 & \mathrm{c_a(t-\delta t, 1)} & \mathrm{c_a(t, 2)} & \mathrm{c_a(t+\delta t, 3)} & \mathrm{c_a(t+2\delta t, 4)} & \ldots \\
1.0 & \mathrm{c_a(t, 1)} & \mathrm{c_a(t+\delta t, 2)} & \mathrm{c_a(t+2\delta t, 3)} & \mathrm{c_a(t+3\delta t, 4)} & \ldots \\
1.0 & \mathrm{c_a(t+\delta t, 1)} & \mathrm{c_a(t+2\delta t, 2)} & \mathrm{c_a(t+3\delta t, 3)} & \mathbf{c_a(t+4\delta t, 4)} & \ldots \\
1.0 & \mathrm{c_a(t+2\delta t, 1)} & \mathrm{c_a(t+3\delta t, 2)} & \mathbf{c_a(t+4\delta t, 3)} & \mathrm{c_a(t+5\delta t, 4)} & \ldots \\
1.0 & \mathrm{c_a(t+3\delta t, 1)} & \mathbf{c_a(t+4\delta t, 2)} & \mathrm{c_a(t+5\delta t, 3)} & \mathrm{c_a(t+6\delta t, 4)} & \ldots \\
1.0 & \mathbf{c_a(t+4\delta t, 1)} & \mathrm{c_a(t+5\delta t, 2)} & \mathrm{c_a(t+6\delta t, 3)} & \mathrm{c_a(t+7\delta t, 4)} & \ldots \\
1.0 & \mathrm{c_a(t+5\delta t, 1)} & \mathrm{c_a(t+6\delta t, 2)} & \mathrm{c_a(t+7\delta t, 3)} & \mathrm{c_a(t+8\delta t, 4)} & \ldots \\
1.0 & \mathrm{c_a(t+6\delta t, 1)} & \mathrm{c_a(t+7\delta t, 2)} & \mathrm{c_a(t+8\delta t, 3)} & \mathrm{c_a(t+9\delta t, 4)} & \ldots \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots
\end{pmatrix} \tag{7.14}$$

The first column contains values of 1.0, because the confidence of true measurement data is 1.0. The second column shows the local confidences when predicting over a single day only. The third column shows the accumulated confidences when predicting over two days, etc. The average accumulated confidence values plotted in Figure 7.1 are the mean values of

each column in Matrix (7.14). The local confidence values were computed using the similarity measure.

The bottom curve of Figure 7.1 repeats the analysis, this time using the proximity measure to compute the local confidence values.

As the similarity measure is more sensitive, the average accumulated confidence values using the similarity measure a a little lower than those using the proximity measure.

Figure 7.2 shows the true average prediction error obtained using FIR (the same curve that was shown already in the top graph of Figure 7.1) plotted together with *normalized* similarity and proximity errors.

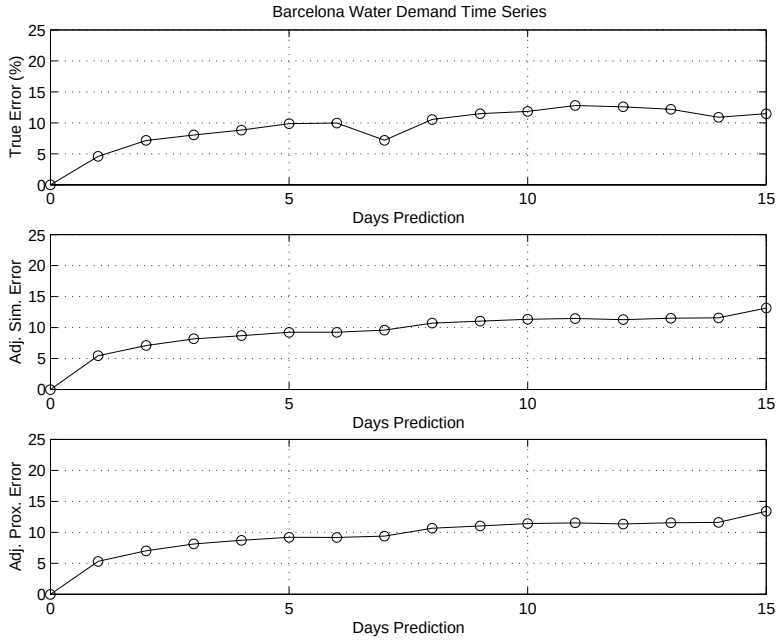

Figure 7.2: Error comparison for Barcelona water demand series.

The similarity error is defined as:

$$err_{\text{sim}} = 1.0 - c_{a_{\text{sim}}} \tag{7.15}$$

and the proximity error is defined as:

$$err_{\text{prox}} = 1.0 - c_{a_{\text{prox}}} \tag{7.16}$$

The normalization is done in the following way. The mean values of the true prediction error over the 15 days is computed as:

$$err_{\text{avg}_{\text{true}}} = \sum_{i=1}^{15} error_i \tag{7.17}$$

and similarly, the average similarity and proximity errors are computed as:

$$err_{\text{avg}_{\text{sim}}} = \sum_{i=1}^{15} err_{\text{sim}_i} \quad ; \quad err_{\text{avg}_{\text{prox}}} = \sum_{i=1}^{15} err_{\text{prox}_i} \tag{7.18}$$

Two ratio factors are computed as follows:

$$k_{\text{sim}} = \frac{err_{\text{avg}_{\text{true}}}}{err_{\text{avg}_{\text{sim}}}} \quad ; \quad k_{\text{prox}} = \frac{err_{\text{avg}_{\text{true}}}}{err_{\text{avg}_{\text{prox}}}} \tag{7.19}$$

The normalized similarity and proximity errors are then defined as:

$$err_{\text{norm}_{\text{sim}}} = k_{\text{sim}} \cdot err_{\text{avg}_{\text{sim}}} \quad ; \quad err_{\text{norm}_{\text{prox}}} = k_{\text{prox}} \cdot err_{\text{avg}_{\text{prox}}} \tag{7.20}$$

Obviously, this approach is flawed, because the true prediction error that is to be estimated by the two confidence errors is used in the computation of the estimate /dots unless it would be possible to come up with an independent way to compute the normalization factors, k_{sim} and k_{prox}, an approach that does not make use of the very data that are to be estimated.

The strong positive correlation between the three curves is evident by naked eye. Hence either of the two confidence errors can be used as a *relative* estimator for the true prediction error. They could even be used as *absolute* estimators, if it were possible to determine the normalization factors independently.

The numerical values of the two normalization factors for Series B are:

$$k_{\text{sim}} = 24.8598 \quad ; \quad k_{\text{prox}} = 36.7448 \tag{7.21}$$

7.3.2 Water Demand of the City of Rotterdam: Series R

Series R had also been introduced in Chapter 4. Figure 4.22 is repeated once more in Figure 7.3.

Dynamic mask allocation was not employed in this case, because the results obtained with the more complex dynamic mask allocation algorithm were no better than those obtained using the optimal mask alone.

Using the same methodology as applied in the case of the Barcelona series, the normalized error curves shown in Figure 7.4 are obtained. The strong

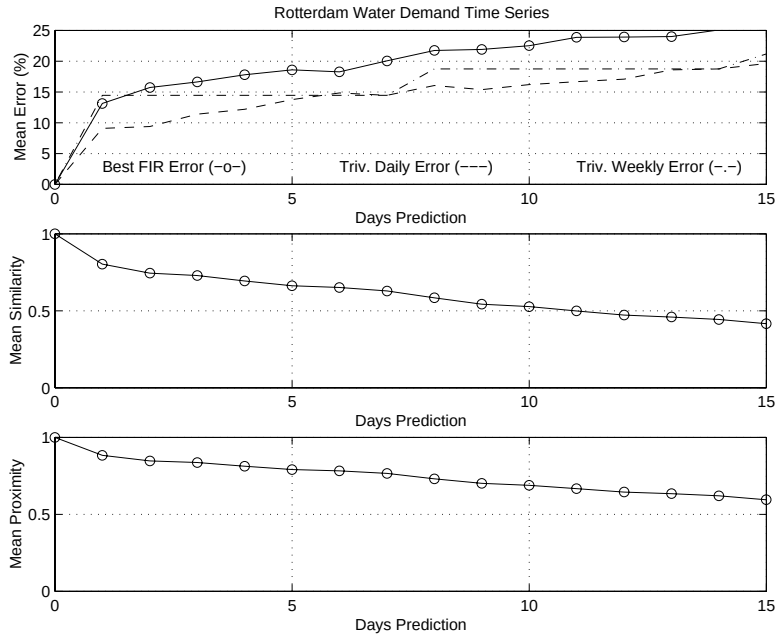

Figure 7.3: Rotterdam water demand multiple–step simulation using FIR.

positive correlation between the true prediction errors and the two confidence errors is evident.

The normalization factors found for Series R were:

$$k_{\text{sim}} = 50.3449 \quad ; \quad k_{\text{prox}} = 77.3699 \tag{7.22}$$

They are about twice as large as for Series B. In order to be able to use the confidence as a quantitative estimate of the true prediction error, the confidence errors, i.e., the *confidence reduction*, should have been about twice as large.

7.3.3 Tucson Weather Prediction: Series T

Series T is a recording of 5000 hours (roughly 7 months) worth of temperature data measured for the City of Tucson. The measurement data are shown in Figure 7.5.

Series T can be characterized as shown in Table 7.1.

Whether a time series is considered stationary or not may depend on the point of view. Series T is stationary when observed over a number of years. It is also stationary, when considered for a small number of days, but it is non–stationary, when considered for a few hours, or for a few months, or for several centuries. In the context of the measured data stream, the series must

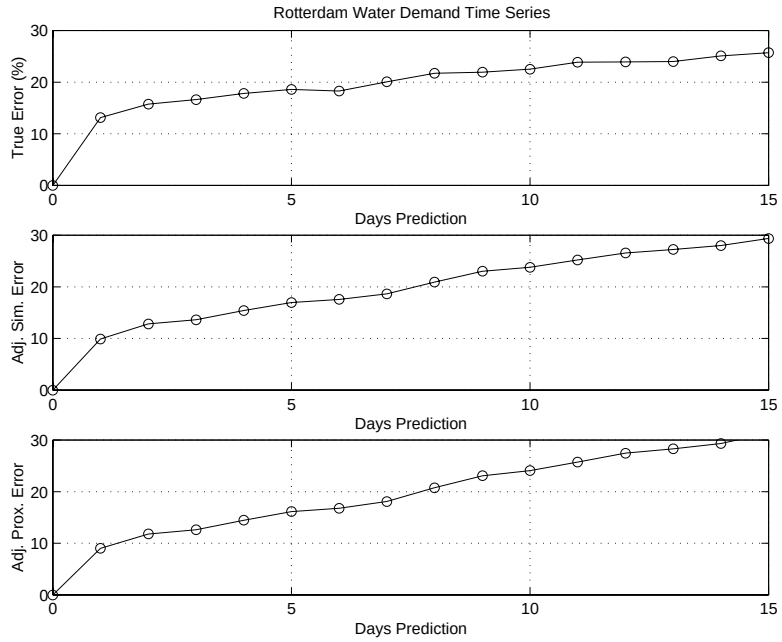

Figure 7.4: Error comparison for Rotterdam water demand series.

be considered non–stationary. It could be considered time–varying because of the trend in the data, but usually, the term "time–varying system" is reserved to denote a system that undergoes more drastic behavioral changes.

The data exhibit a strong daily cycle. The auto–correlation function is presented in Figure 7.6. Hourly measurements were available for the entire year 1995, though only a subset of the available data were used in Series T.

This time series is of particular interest, because there exists a rich literature about weather prediction and the (usually quantitative) models used for it. It is well established that a prediction over about five days is feasible from local data, whereas a longer–term prediction will not work due to the chaotic nature of the underlying physical system. It is to be expected that the simple FIR model used in this chapter will do a much poorer job than the sophisticated partial differential equation models discussed in the open literature, as it only takes into account previous ambient temperature values, ignoring other important factors such as cloud cover, humidity, sky radiation, and the effective temperature of the night sky, for which measurement data are also available.

As a base line for comparison, two trivial predictors, an *hourly trivial predictor* and a *daily trivial predictor* were simulated in parallel with FIR.

5000 of the available data points were used as training data, whereas another 128 data points were used for testing. Because of the 24–hour cycle,

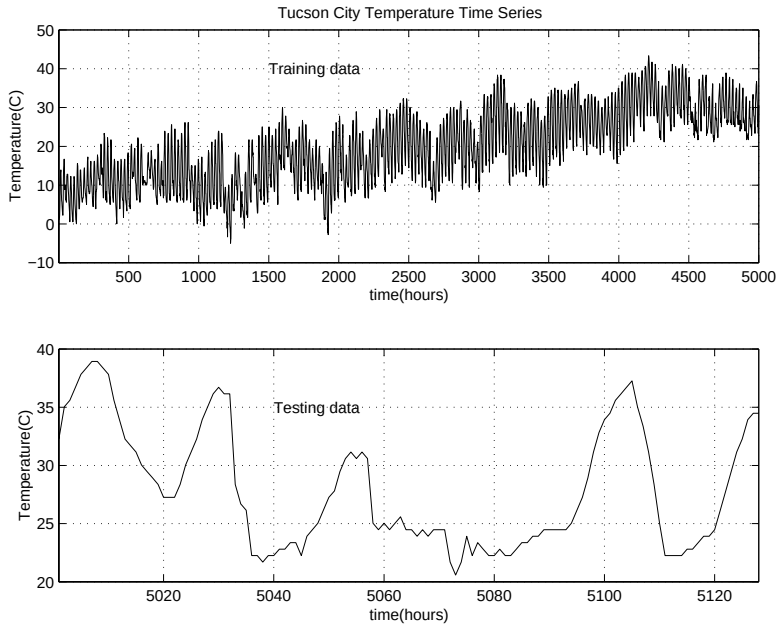

Figure 7.5: Training and testing data for Tucson weather prediction.

it was decided to use a mask depth of 50, so that FIR would find two entire days in the data covered by the mask.

The optimal model proposed by FIR is the following:

$$\begin{array}{cc} & y \\ t-48\delta t & \left(\begin{array}{c} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ 0 \\ 0 \\ 0 \\ -3 \\ +1 \end{array}\right. \\ t-47\delta t & \\ \cdots & \\ t-25\delta t & \\ t-24\delta t & \\ t-23\delta t & \\ \cdots & \\ t-2\delta t & \\ t-\delta t & \\ t & \end{array} \tag{7.23}$$

Since there exists a strong 24–hour cycle, FIR proposes to use the values 24 and 48 hours back for its predictions. The selection is reasonable.

Hourly predictions over up to 50 hours were performed, i.e., a prediction table with 51 columns and 128 rows was chosen.

Figure 7.7 compares, in its top portion, the prediction errors resulting from the use of FIR on the one hand and of the two trivial predictors on the

Table 7.1: Classification of Time Series T

natural	T	synthetic	
stationary		non–stationary	T
time invariant	T	time varying	
low dimensional		stochastic	T
clean		noisy	T
short		long	T
dormant		active	T
documented	T	blind	
linear		non–linear	T
scalar	T	vector	
single recording	T	multiple recordings	
continuous	T	discrete	

other.

FIR does not accomplish much. For the first five hours, FIR is outperformed by the hourly trivial predictor, thereafter it is outperformed by the daily trivial predictor. Only during a few hours, around the 24 hour prediction, does FIR slightly outperform its two trivial competitors.

The lower two graphs of Figure 7.7 show the two accumulated confidence measures. In spite of the sobering prediction quality, FIR's confidence in its own predictions is extremely high.

Using the same methodology as applied in the case of the Barcelona and Rotterdam series, the normalized error curves shown in Figure 7.8 are obtained. As before, a strong positive correlation between the true prediction errors and the two confidence errors can be observed.

The normalization factors found for Series T were:

$$k_{\text{sim}} = 195.8103 \quad ; \quad k_{\text{prox}} = 252.6687 \tag{7.24}$$

Where does this high confidence in FIR's predictions originate from? In all three cases, FIR proposed a mask of complexity 4 as the optimal mask, i.e., the optimal masks of each of the three time series are characterized by three m–inputs that were discretized into three levels. Thus, the minimal number of records needed for making a model is:

$$n_{\text{rec}} \geq 5 \cdot 3^3 = 135 \tag{7.25}$$

The actual number of training data records used in the three series are shown in Table 7.2.

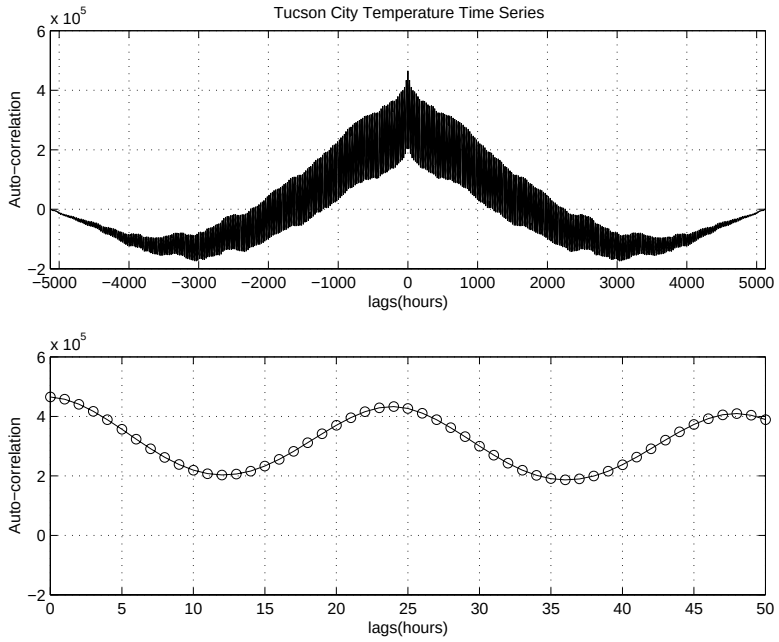

Figure 7.6: Auto–correlation of Tucson weather data.

Table 7.2: Training data for Series B, R, and T

Barcelona	570
Rotterdam	3500
Tucson	5000

Due to the narrow peaks in the Barcelona series, even 570 data records were not enough to characterize well the weekends. Thus, the theoretical minimum of 135 data records is clearly insufficient.

Because of the more stochastic nature of Series R and T, more training data records were used. In the case of Series T, the data were furthermore oversampled, which led to the need for a very deep mask.

The abundance of training data led to optimistic confidence estimates. The confidence measures contain two separate parts: a measure of the distance (or dissimilarity) of the five nearest neighbors from the testing data record in the input space, and a measure of the dispersion among the five nearest neighbors in the output space.

By adding additional training data to the training data set, the former of these two measures can be made arbitrarily small, leading to a complete confidence in the *quantity* of available training data. This is what happened

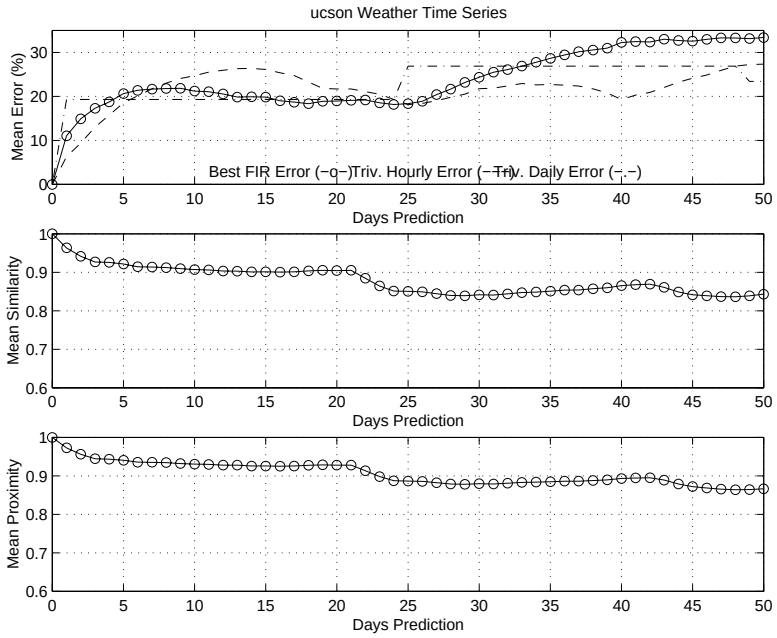

Figure 7.7: Tucson temperature multiple–step simulation using FIR.

both in the case of Series R and in that of Series T.

Oversampling leads to an optimistic estimate of the latter of these two measures as well. If a time series is oversampled, the five "nearest neighbors" may in fact not be five truly different neighbors, but only five recordings of one and the same neighbor. Obviously, if the five nearest neighbors were recorded during neighboring sampling instances, their outputs will be correlated, which reduces the dispersion among them. This is what happened in the case of Series T.

It is therefore reasonable to assume that the numerical values of the normalization factors depend on the ratio of the available training data records to the minimally required number of training data records, and also on the ratio of the chosen sampling rate to the optimal sampling rate.

The author stipulates that formulae estimating the values of the two normalization factors can be found; however, this will require additional experimentation. Additional time series need to be analyzed, and also, individual time series, such as the Tucson temperature series, need to be analyzed under varying experimental conditions:

1. by varying the number of training data records

2. by varying the sampling rate

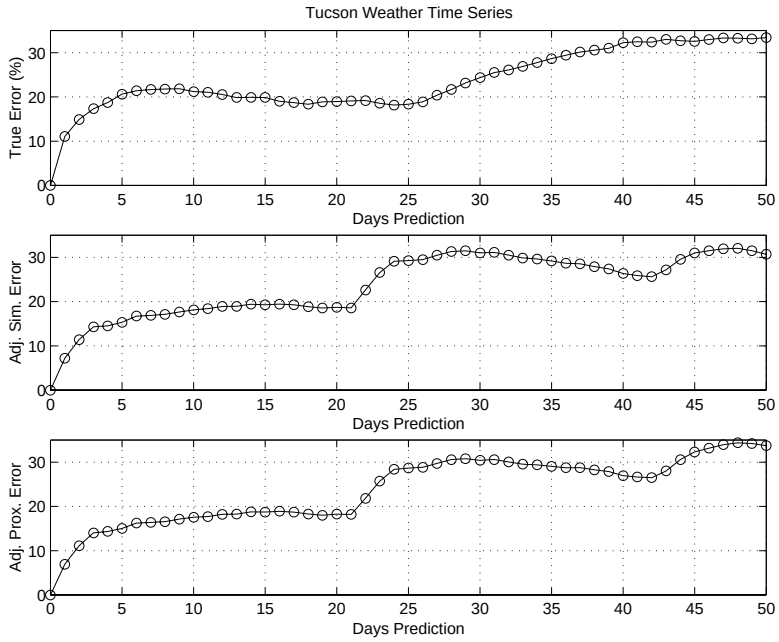

Figure 7.8: Error comparison for Tucson temperature series.

3. by varying the number of discretization levels.

However, such an effort was not undertaken as part of the research described in this dissertation.

Figure 7.9 compares the one–hour prediction, the 24–hour prediction, and the 48–hour prediction of FIR with that of its two trivial competitors.

In the one–hour prediction, the trivial hourly predictor and FIR both predict well, whereas the trivial daily predictor performs much poorer. In the 24– and 48–hour predictions, the hourly and daily trivial predictors are identical, and the performance of all three predictors is about equal in quality.

Time series relating to weather prediction are a fascinating research subject, because there exists a wealth of knowledge about weather prediction. The results shown in this dissertation are only a first step in this direction.

A natural next step would be to consider all the measurement data that have been recorded, namely:

1. temperature
2. humidity
3. cloudiness
4. solar radiation

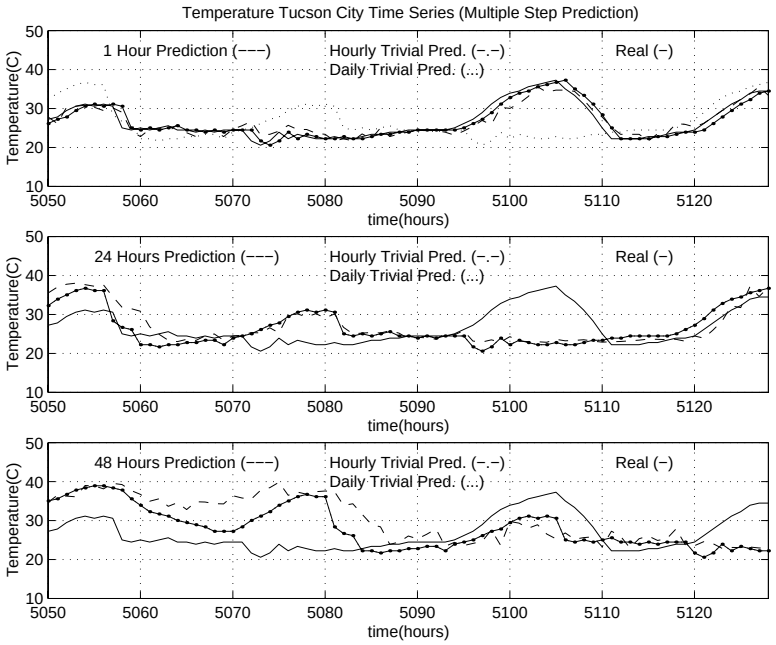

Figure 7.9: One–hour, 24–hour, and 48–hour predictions of Tucson temperature series.

5. radiation of the night sky

to form a multi–variate time series, and predict all of these variables in parallel. The correlations among these variables should help improve the forecasts.

A first attempt at tackling this problem was carried out by a senior project student at the University of Arizona (Chabot 1998). However, due to the limited time available to the student for her project, the results obtained are inconclusive. The programs need to be refined, and more simulation experiments need to be performed.

A next step would be to exploit the structural knowledge available from quantitative weather models to structure the FIR model using the structured approach proposed by M.Moorthy (Moorthy 1999). Natural state variables of the weather model are:

1. the amount of sensible heat stored in a unit volume (related to the temperature)

2. the amount of latent heat stored in a unit volume (related to the humidity)

3. the amount of convective flow due to wind (kinetic energy)

Natural driving functions are the solar and night–sky radiations. Natural outputs are the temperature, the humidity, and the barometric pressure.

At any point in time, the extraneous driving functions are estimated first, based on their own past. Luckily, the radiation patterns are highly regular, and can therefore be predicted well. Then, the current values of the state variables are predicted based on their own past, on the current and past values of the driving functions, and on the past values of the outputs. Finally, the current values of the outputs are estimated based on the current and past values of the driving functions and state variables, as well as on their own past.

It would be furthermore interesting to compare these results against the best results obtainable from quantitative weather models that are simulated by numerically solving the Navier–Stokes equation for a geographic region with a given discretization. It would be highly interesting to know how well FIR can match the numerical simulation results in quality.

7.4 Conclusion

In this chapter, it was shown that the accumulated confidence measure can be used as a *relative* estimator of the magnitude of the average prediction errors. This means that, for any given time series, larger confidence values are an indicator for an increased likelihood of smaller prediction errors. However, the confidence values cannot be compared across different time series, or across different experiments (different number of training data, different sampling rate, different number of discretization levels) for the same time series, because the absolute value of the confidence measures depend on the experimental setup as much as on the time series itself.

Once the accumulated confidence plot has been obtained and the corresponding normalization factor has been estimated, it is possible to determine how far into the future a prediction can be made. For example in the case of Series B, testing data can be used to establish a normalization factor of $k_{\text{sim}} = 25$ for the similarity measure. If a 5% error is tolerated, then the accumulated confidence must be limited to 80%. Predictions that carry an accumulated confidence value of below 80% must consequently be rejected. This corresponds, for practical purposes, to a predictability horizon of one day. On the other hand, if prediction errors of up to 10% are acceptable, then the accumulated confidence must be limited to values above 55%. Predictions with a lower accumulated confidence value must be rejected. For practical purposes, this results in a predictability horizon of roughly eight days.

Chapter 8

Conclusions

The author's involvement with Fuzzy Inductive Reasoning began in the spring of 1994, when she signed up for a *curso de doctorado* with *Professor Rafael Huber*. She was fascinated with the possibilities that the methodology seemed to offer in terms of its ability to make predictions about the future, in terms of its support for studying and analyzing the unknown, in its ability to support exploratory research.

During the summer of that year, the author met *Professor François Cellier* who agreed to work with her toward a Ph.D. degree. Dr. Cellier proposed to look at the self–assessment capabilities of FIR and come up with a formal analysis and possibly improvement of the quality of predictions made. He also suggested the analysis of time series, because they would provide for a framework, where FIR could be compared against other competing forecasting methodologies. Because of her earlier involvement with time series and their prediction, *Dr. Gabriela Cembrano* was asked whether she would co–direct the dissertation, a task that she gladly accepted.

The author liked the proposed topic, and was thrilled with the prospect of being able to *improve* the forecasting capabilities of the methodology. She studied carefully how FIR makes its predictions, and devised means for how the approach could be improved.

Her first ideas centered around the introduction of *redundancy* into the methodology, in the hope that redundant information would help sharpen the forecasting power of the methodology. In the spring of 1995, a first article had been written, extending the *Fuzzy Inductive Reasoning (FIR)* methodology to the *Causal Inductive Reasoning (CIR)* approach (Cellier and López 1995). CIR differed from FIR in that it added a fourth piece of information to the qualitative triple: a *qualitative gradient*.

The idea was rather straightforward. CIR would make predictions not only about the value of a variable, but also about its tendency to either

increase or decrease. The hope was that this would help the methodology with distinguishing between good and bad forecasts, as good forecasts would be *consistent* in their predictions. A forecast that showed e.g. a growth in value, yet predicted a negative gradient would be filtered out as a bad forecast.

It did not work. The predictions made by CIR were not significantly better than those made by FIR — they were only slower. FIR is so good at picking out patterns that the introduction of redundancy did not improve much its forecasting capabilities that are already close to optimal. The author had to learn the hard way that it sometimes helps to first fully understand the traits of a methodology, before trying to improve upon it.

In the sequel, the author concentrated more on the other part of the project: the analysis of the self–assessment capabilities of FIR. The formulae for the *confidence measures* were developed that had been introduced in Chapter 5 of this dissertation, as well as the *error formula* proposed in Chapter 3, a formula that had to be modified many times before it became robust enough to yield meaningful results in most (though still not all) applications. By the spring of 1996, the next two publications were ready. These talked about the confidence measures (Cellier *et al.* 1996) and their application to the prediction of time series (López *et al.* 1996).

The author then continued to work on development and refinement of the *accumulated confidence measures* needed to estimate the forecasting horizon, i.e., the *horizon of predictability*, as presented in Chapter 7.

It was not without irony that a contribution to the forecasting capabilities of the FIR methodology finally came, when it was least expected, namely as a by–product of developing the confidence measures. By making multiple predictions in parallel using different suboptimal masks and comparing their confidence values, the quality of FIR predictions could indeed significantly be improved. This aspect of her research has been captured in Chapter 6 of the dissertation.

The primary contributions of this dissertation are now summarized.

Chapter 3 offers two primary contributions: a new formula to assess the error of predictions of a univariate time series, and the discovery that FIR filters out what it considers to be noise.

The new error formula punishes to equal parts: deviations of the mean of the prediction from the mean of the series, a difference between their standard deviations, differences in the absolute errors between the two trajectories after normalization of their means and standard deviations, and differences in their shape, i.e., the so–called dis–similarity error.

The longest chapter of this dissertation, Chapter 4, provides the backbone of the analysis of FIR's forecasting capabilities. In this chapter, FIR is

compared against an extensive, though not exhaustive, number of alternate approaches in its capability of making forecasts. Two time series, relating to the prediction of water consumption in the cities of Barcelona and Rotterdam, were used to make this comparison.

One of these series, Series B, was sufficiently deterministic to make a meaningful prediction possible. The other series, Series R, was much more stochastic, and did not contain enough regularity to allow for a meaningful prediction beyond that provided by the trivial predictor. FIR was among the best techniques in predicting Series B, and it concluded, with all the other techniques, that a prediction of Series R was hopeless.

Previous research on Series B (Quevedo *et al.* 1988) had produced a Box–Jenkins (ARIMA) model of the water demand, which is currently in use in the city's water distribution management system. This model is considered the most important reference for comparison for the forecasting results of the Barcelona demand series, and the fact that both methodologies, Box–Jenkins and FIR, produce forecasting errors that do not differ significantly is considered an important asset of FIR. FIR provides the model *automatically* whereas the Box–Jenkins methodology requires a significant development effort as well as knowledge about the nature of the process from which the series was derived. Additionally, the need for interventions in the ARIMA methodology is, to some extent, overcome by the use of the "five–nearest–neighbor" rule that FIR employs in its predictions, at least for interventions that extend over more than one day in a row.

FIR was found to exploit the available knowledge *consistently* and *reliably*, and does so in a predominantly algorithmic fashion, i.e., setting up a FIR model is fast and painless in comparison with other sophisticated techniques, such as the Box–Jenkins and ANN methodologies. Furthermore, its self–assessment capabilities give FIR an advantage over the competition, the importance of which cannot be overestimated.

The primary contributions of Chapter 5 are the newly introduced confidence measures. Although it is not possible to find a deterministic estimate for the prediction error, as was shown in Chapter 5, the confidence measures provide at least a statistical estimate for the quality of the prediction. The self–assessment capability of FIR is easily its most significant characteristic. Its importance cannot be overestimated. By developing these confidence measures and implementing them in the SAPS–II software, our current implementation of the FIR methodology, a facet was added to the methodology that significantly increases the value of the tool.

Chapter 6 demonstrates how the previously introduced confidence measures can be used to significantly improve the quality of forecasts made by FIR. To this end, several suboptimal masks are used to make, in parallel,

forecasts of the same time series. Each of the forecasts is accompanied by an estimate of its quality. In each step, the one forecast is kept as the true forecast to be reported back to the user that shows the highest confidence value.

Chapter 7 finally deals with the effects of error accumulation across multiple steps of prediction in a simulation mode, whereby previous predictions are being used as data inputs in making subsequent predictions. The data contamination problem associated with such iterative predictions is discussed, and a set of formulae has been devised to estimate the effects of data contamination on the accumulated confidence over multiple prediction steps.

Has this research effort been *completed*? In the views of this author, engineering research is *always* open–ended. It is never complete. Contrary to the pure sciences, such as mathematics, where it may be possible to prove a lemma that settles an open question once and for all, in engineering, research never settles any question permanently, except in a negative sense. If a hypothesis that has been posited could be disproved by finding a counter–example, then this settles the question once and for all. However, positive contributions are invariably only milestones in an open–ended search. They are pearls along an infinite necklace, and each new pearl only opens the prospect of finding the next one by proceeding further along the same path.

Two related research efforts have already been started by the author of this dissertation. The findings, though preliminary in nature, are reported in Appendices A and B of this dissertation.

Appendix A discusses the use of time–series predictors in the design of *smart sensors* with look–ahead capabilities that may in the future be used in the monitorization of complex engineering processes, such as nuclear power plants (de Albornoz 1996). The idea behind this application is simple: if a sensor with look–ahead capability can anticipate the crossing of a critical threshold, it may issue an early warning that might enable the plant operator to do something about the problem before it ever occurs.

Appendix B introduces a new class of predictive controllers, coined *signal predictive controllers* that make use of smart sensors of the class introduced in Appendix A to improve the control performance of feedback control systems.

A practical issue that the author has pondered for several years, but never found the time to pursue, is the following. Most qualitative simulation engines, such as QSim (Kuipers and Farquhar 1987), do not predict a single trajectory (or episode). Instead, they predict an envelope of all trajectories (episodes) that are consistent with the available knowledge. It would be useful to implement in SAPS–II a tool for pursuing multiple predictions in parallel, such that an envelop of either the *possible* or the *probable* predictions

can be obtained.

One interesting theoretical question that would be well worth pursuing is the following. Is it possible to come up with a *quantitative* analysis determining how close a prediction algorithm came with respect to exploiting all of the information available in a given time series?

The question seems a difficult one, but it is not hopeless. To this end, it would be interesting to analyze yet another class of competitors, the so–called *general predictors*. These predictors are based on a concept that was developed by Claude Shannon (Shannon 1951). Shannon performed an experiment, in which a human subject was asked to guess the next letter in an English text. If the guess was correct, the subject was told so, otherwise, the subject was allowed to make further guesses, until the next letter had been guessed correctly. Shannon reported the outcome of the experiment as shown in Figure 8.1.

Underneath each letter, Shannon recorded the number of guesses it took, before the subject came up with the correct letter. Suppose that the subject is highly systematic, and therefore guesses always in a purely algorithmic manner. In this case, it would not be necessary to record the letters at all. The numbers underneath them would be equivalent.

This idea leads to a class of *data compression* algorithms, called *general compressors*. General compressors sound outlandish at first, but they are meanwhile widely used as data compressors for images and sounds on a computer, to reduce the size of files to be transferred to a minimum.

It is interesting to note that a *theoretical limit* of data compression can be computed, i.e., for any real compression, it is possible to know how close to the theoretical limit it is.

In the limit, the data file will invariable have to look like uncorrelated white noise. There is no longer *any* information contained in the data at all. The entire information now is in the *key*, i.e., in the data compression/un–compression algorithm.

The problem of ideal (general) data compression is related to that of ideal (general) *data encryption*. Obviously, an ideally compressed data file can be transmitted without risk of ever being deciphered, as long as the key is kept secret.

How is this problem related to that of *prediction*? Already the original work of Shannon hints at a close relationship. After all, the experiment was about predicting a time series. If a time series has been ideally compressed, the compressed version will look like uncorrelated white noise. Hence it is possible to extend the data file by appending to it any uncorrelated white noise, then interpret this extension as a compressed signal, and use the key to un–compress it. This leads to a prediction of the series. By repeating

the experiment with different noise extensions, different predictions result that are all feasible in the context of the available knowledge about the time series.

Finally, the engineering applications of the methodology developed in this thesis, as described in Appendices A and B, have barely been touched. Much more research will be needed to bring these ideas to fruition. An entire Ph.D. dissertation could be written on designing a robust and reliable *Signal Predictive Control* algorithm alone.

Luckily, not all things are predictable. As long as there remain open questions, there is hope. As long as there is hope, the world moves on. As long as the world moves on, there remains something to be predicted.

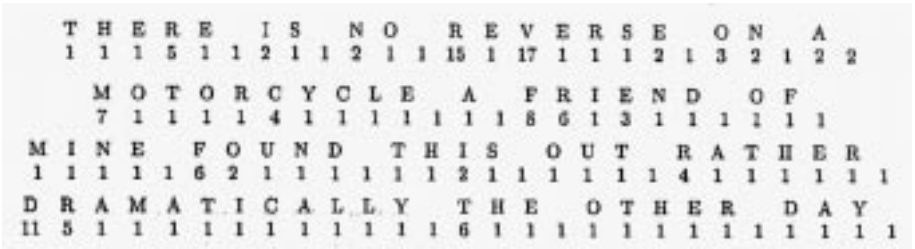

Figure 8.1: Shannon experiment

Appendix A

Early Warning Using Smart Sensors with Look–Ahead Capabilities

A.1 Introduction

The average complexity of engineering systems in use has steadily grown over the years. Whereas fifty years ago, a homeowner would simply light a fire in the open fire place when it got too cold in the house, and otherwise control his body temperature by wearing an extra jacket, modern houses are now full of elaborate control circuitry. Heating systems are fully automated and controlled by one or multiple thermostats that regulate the temperature in the house. These thermostats are furthermore programmable in such a way that the set point temperature can be reduced during the day time, when everyone is at work, or during the night, while everyone is asleep.

Modern cars are equipped with computers that control the engine in many subtle ways. When the computer registers a potential problem, it alerts the driver and suggests that the car be brought to a mechanic. The mechanic then connects the computer inside the car across the Internet to the master computer of the car manufacturer, and receives immediately a printed report of which parts need to be exchanged.

One of the most complex engineering systems ever built is Biosphere 2, a closed–ecology environment located 50 km north of Tucson (Mitsch and Marino 1999). In this system, there are controllers for the temperature, humidity, and air pressure in each of five biomes. The biosphere system contains 1800 sensors whose values are recorded on the average once every 15 minutes to monitor the state of the biosphere.

Due to the interconnections of subsystems, the malfunction rate of a complex engineering system grows at least quadratically in the number of components that the system is composed of (de Albornoz 1996). The purpose of the control systems is not only to keep the variables of the system within prespecified ranges, but also to prevent the system from malfunctioning.

The most critical components of a control system are its *sensors*. Most control laws are actually quite simple. However, the success of implementing such a control law depends heavily on the quality of sensory information available about the state that the process to be controlled is in.

In order to improve the quality of information available, many modern control systems employ so–called *smart sensors*. Smart sensors are sensory data processing systems that interpret the raw data obtained by the sensors, and present the control algorithm with prefiltered rather than raw sensory information. They operate in part by means of *sensor fusion*, i.e., they collect correlated information from multiple sensors, and make use of the redundancy contained in these data streams to calculate a single much cleaner signal to be forwarded to the control algorithm. They may also employ other techniques, such as look–ahead algorithms, i.e., real–time simulations that predict ahead of time an expected future value of a sensor on the basis of its current and past values.

Any control architecture contains essentially three parts:

1. the *sensory system* that is responsible for recording the current state of the system to be controlled,

2. the *control algorithm* that makes use of the sensory information for determining appropriate control actions, and

3. the *actuators* that translate the control actions into physical signals that can be applied to the control inputs of the plant.

Whether "smart sensors" are considered part of the sensory system or part of the control algorithm is a matter of personal taste. The name "smart sensor" suggests that traditionally they have been associated with the sensory system. There is a rationale for this decision: most smart sensors only concern themselves with the question of where the system to be controlled *currently is*, whereas the task of the control algorithm is to determine where the system *is going*.

This appendix, however, deals with a class of smart sensors that concern themselves with the question of where the system is going, by making predictions of future values of sensory information based on recordings of their current and past values.

Humans constantly make use of *state predictions* when they reach control decisions. A driver of a car who sees a ball rolling out onto the street from behind a parked vehicle will hit the brakes at once, not because the ball would pose any serious obstacle to the car, but because he or she knows that balls do not move on their own, and therefore expects a small child to follow the ball out onto the street shortly. A capitalist will sell stocks on the stock market when he or she expects the value of the stocks to decrease, and will buy stocks, when it can be expected that the stocks are going to gain value.

Smart sensors with look–ahead capabilities tie neatly into the framework of this dissertation, since state prediction relates directly to the question of time–series analysis and forecasting. Previous research published about this problem made use of either neural networks or statistical approaches, such as principal component analysis (Qin 1998). In this appendix, FIR will be offered as an alternative technique for such smart–sensor designs.

A.2 Early Threshold Detection

The aim of this appendix is to analyze and discuss the possibilities of designing smart sensors that can provide a controller with an early warning about a threshold to be passed in the near future. The rationale behind the research is straightforward: by the time a sensor detects that a threshold has been passed, it may be too late to do anything about the problem, because, due to the inertia in the system, control actions may not take immediate effect. Traditionally, such systems had to set their thresholds more narrowly in order to provide the controller with an early warning. However in this way, the entire allowed range of values is not exploited, which may reduce the performance of the system.

Figure A.1 illustrates the concept of a narrowed–threshold design. Since all physical systems are characterized by a finite bandwidth and finite energy, state variables can never jump, and therefore, a reduction of the range of allowed sensor values necessarily offers an early warning capability.

Unfortunately, this approach may limit the performance of the system. For example, if it would be safe to drive along a road with 60 km per hour (upper limit), the reduction may result in a reduced maximum tolerated speed of 50 km per hour. If it would be safe to follow another car at a distance of 100 m (lower limit), it may now be necessary to keep a distance of 150 m, etc.

To prevent this undesirable reduction in performance, some systems make the range reduction depend on the gradient with which the threshold is being approached. For example, the maximum allowed speed may be set

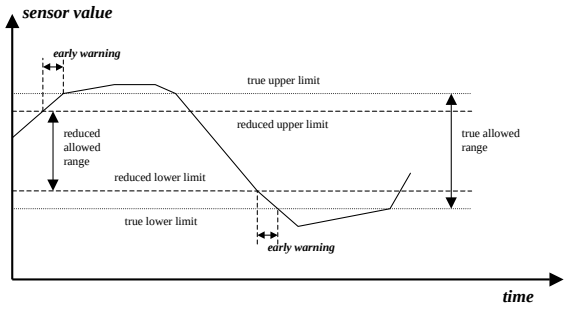

Figure A.1: Early threshold detection by reduced sensor value range.

to (60 km/h $- k \cdot a$), where a is the current acceleration of the car, and k is a proportionality constant. In classical control terminology, this approach corresponds to introducing a D–term into the control scheme. This technique shall therefore be coined the *PD–approach*.

In this dissertation, another route shall be taken. A FIR model predicts future values of the sensory signal ahead of time. If such a prediction reaches the true threshold, a warning is issued claiming that the threshold is predicted to be reached in t time units. By providing the controller with an early warning, the controller may still have time to calculate a new control action that will prevent the threshold from ever being reached.

If the prediction is imprecise, two types of errors may occur:

1. The prediction may issue an early warning because it foresees that the signal will cross the threshold, although in reality, this would not happen under the current control strategy.

2. The prediction does not foresee any problem, although the signal in fact will cross the threshold after some time.

The first type of error will lead to the use of an overly conservative control strategy (similarly to a reduced–range approach), whereas the second type of error leads to a reduction in the early warning time available to the controller (eventually, as the signal approaches the threshold, a warning will be issued, but it may arrive too late to still be useful).

The FIR prediction approach is similar in its effects to the aforementioned PD–approach, but may work a little better, because its prediction of the signal takes the non–linearity of the system generating the signal into account, which is outside the capabilities of the PD–approach.

Clearly, it will be important to know how accurate the predictions are, and how far into the future forecasts can be made. Therefore, the results presented in Chapters 5 and 7 of this dissertation are highly relevant to the investigation reported in this appendix.

A.3 Application: The Copper Bar

To validate the proposed approach, the following system was used. A copper bar of length 1 m and radius 1 cm is connected at one of its ends to a voltage source of 220 V. The other end is grounded. Hence a current flows through the bar that heats the bar to a certain temperature above the ambient temperature. If the ambient temperature were constant, the temperature of the bar would also reach a constant value. However, the bar interacts with its environments by means of radiation, i.e., if the ambient temperature rises, so does the temperature of the bar, and vice–versa.

Figure A.2 shows the bond graph model used to describe this system. The model follows the methodology outlined in Chapter 8 of (Cellier 1991).

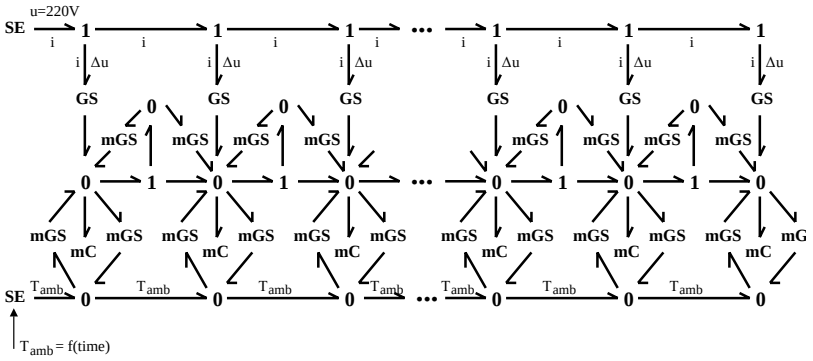

Figure A.2: Bond graph model of a copper bar.

The 0–junctions in the center represent the temperature values at each of 10 segments. The mC elements attached to each of these junctions represent the heat capacities associated with these segments. The 1–junctions

in between these 0–junctions calculate the temperature differences between neighboring segments, which is represented by the 0–junctions just above. The temperature difference ΔT_i multiplied by the entropy flow through the segment represents the amount of heat flow added by means of thermal conduction. The two mGS elements emanating at the 0–junction storing ΔT_i feed this added entropy back into the bar at the 0–junctions to the left and to the right.

The SE element at the top of Figure A.2 represents the voltage source. The 1–junctions to its right represent the current flowing through the bar. At each segment, the voltage drops by Δu Volts. The product $\Delta u \cdot i$ is the electrical power that gets converted to heat in each of the segments. This conversion is symbolized by the GS elements. The entropy generated by these elements is delivered back to the 0–junctions representing the temperature of each segment.

The SE element at the bottom of Figure A.2 represents the ambient temperature, which is assumed to be a function of time, i.e., this SE element is a non–linear effort source. The 0–junctions to its right symbolize the fact that the ambient temperature is the same all along the copper bar. The mGS elements emanating and ending in these 0–junctions represent the radiative flow between the copper bar and its environment. The convective flow was neglected in this model as a phenomenon that is of second order small, an assumption that is correct as long as there is no forced air flow around the copper bar.

The bond graph model was encoded in Dymola (Dynasim 1996). The Dymola compiler was then used to translate the model into ACSL (MGA 1998) for simulation. The simulation results were then converted to a Matlab (MathWorks 1997) matrix for use by FIR.

To make the example interesting, a disturbance had to be introduced that modifies the ambient temperature as a function of time. In lack of any better disturbance function and since the example is synthetic anyway, Series T was used once again, however this time, it was used as a disturbance function, rather than as a time series to be predicted. Hence the ambient temperature of the copper bar changes with the same patterns as the temperature in Tucson. However, in order to make the time constants of the disturbance (the ambient temperature) commensurate with those of the system (the thermal time constants of the copper bar), Series T was compressed in time by a factor of 36. Hence a "daily" temperature cycle is now completed within 40 minutes.

The resulting time series, representing the temperature at the far end of the copper bar, i.e., where it is grounded, is shown in Figure A.3. This time series shall be called Series U, denoting the *uncontrolled copper bar*. 6000 data

points were collected, corresponding to roughly 7 days of simulated time with a sampling rate of 100 sec. The first 5000 of those data points were used for model identification, whereas the following 200 data points were used for model validation.

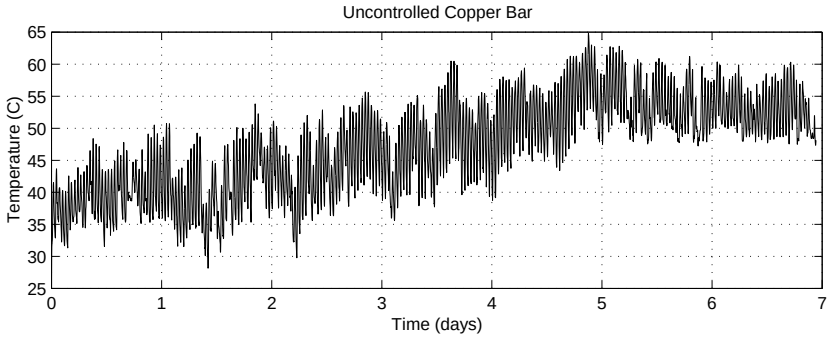

Figure A.3: Temperature of the copper bar.

Series U has essentially the same characteristics as Series T, except that it ought to be classified as "synthetic" rather than "natural."

Table A.1: Classification of Time Series U

natural		synthetic	U
stationary		non–stationary	U
time invariant	U	time varying	
low dimensional		stochastic	U
clean		noisy	U
short		long	U
dormant		active	U
documented	U	blind	
linear		non–linear	U
scalar	U	vector	
single recording	U	multiple recordings	
continuous	U	discrete	

The model identified by FIR is the same as for Series T, i.e.

$$\begin{array}{ll} time & u \\ t-47\delta t & \left(\begin{array}{r} -1 \\ 0 \\ 0 \\ 0 \\ -2 \\ +1 \end{array}\right) \\ t-46\delta t & \\ \cdots & \\ t-2\delta t & \\ t-1\delta t & \\ t & \end{array} \tag{A.1}$$

Figure A.4 shows the one–step and two–step predictions of this time series. The reader may notice the usual lag of the prediction behind the true value.

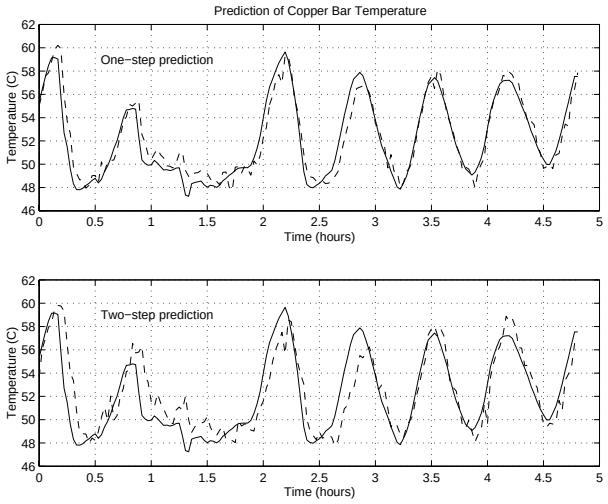

Figure A.4: One–step and two–step predictions of copper bar temperature.

It is interesting to notice that, in Chapter 7, it was shown that the FIR predictor does not outperform the trivial hourly predictor for the first few hours. Yet for the purpose of an early warning, the trivial predictors are totally useless, because they do not look ahead at all. Thus, it will be interesting to see whether the FIR predictor indeed does have a look–ahead capability.

Figure A.5 shows the same curves once more as Figure A.4. However, for the task at hand, it is more useful to now plot the prediction *at* the time it is made, rather than *for* the time it is made. Superimposed with the graph are the upper and lower thresholds, which were set at 58°C and 50°C, respectively.

Figure A.6 shows a blow–up of Figure A.5 for the first hour.

It can be seen that the first threshold can indeed be predicted fairly well. A one–step (100 sec) prediction leads essentially to an early warning of

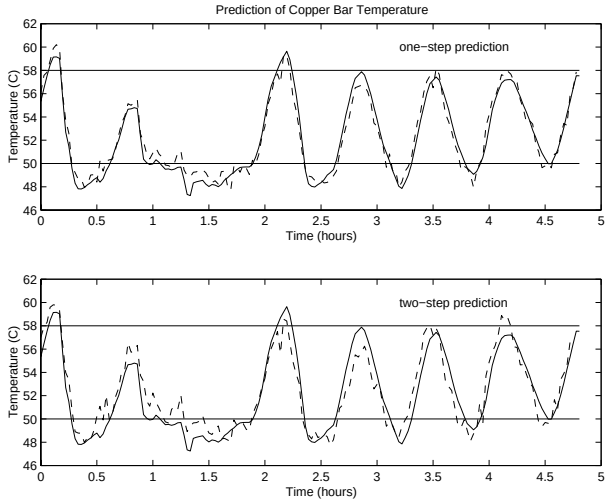

Figure A.5: One–step and two–step predictions of copper bar temperature.

100 seconds, and a two–step (200 sec) prediction leads to an early warning of 200 seconds.

Unfortunately, already the second threshold lies beyond the horizon of predictability, as the prediction lags behind the true event, i.e., is essentially useless. The reason for this discrepancy is that the estimator slightly overestimates the temperature as the previous cycles were a little warmer. Therefore, the upper threshold can be estimated better than the lower threshold during the window shown in Figure A.6.

However, this is not always the case. The lower thresholds at 3.1 hours and 3.8 hours of Figure A.5 are being predicted. The upper threshold at 4.1 hours is an error of type 1, as the predictor predicts a threshold crossing that, in reality, never takes place.

Figure A.7 shows three–step up to five–step predictions.

A five–step prediction predicts 500 seconds, i.e., more than 8 minutes ahead. Yet, it does not predict the first threshold passing until about 4 minutes prior to the true event, because the horizon of predictability is drawing close. The horizon of predictability varies between 0 minutes (when no early prediction is possible) and about 10 minutes.

It would also be possible to combine the FIR approach with a PD–approach, and issue an early warning whenever either of the two approaches indicates a potential problem.

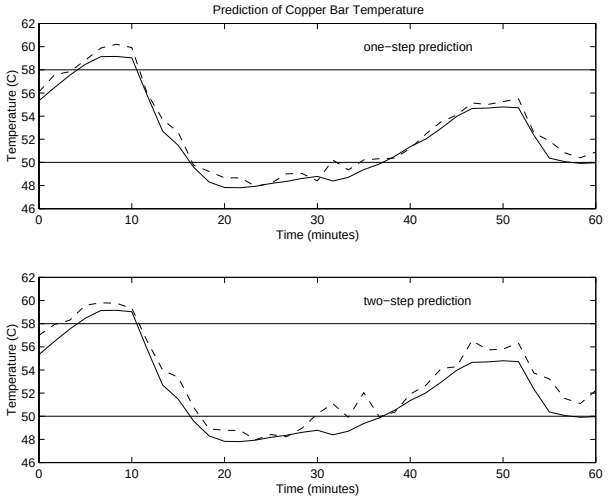

Figure A.6: One–step and two–step predictions of copper bar temperature.

A.4 Conclusions

In this appendix, it was shown that a FIR predictor can be used to implement a smart sensor with look–ahead capability to provide early warning of an impending threshold crossing. Although the FIR predictor did not outperform the trivial predictor, which is useless as a look–ahead tool, for the time series under investigation, FIR indeed could be used as a prediction tool. The reason is that the error measure used in the comparisons of this dissertation measures a different type of error that is not necessarily indicative of success or failure for the task at hand.

The results presented in this appendix must be considered preliminary, because the method was only applied to one single example, moreover a synthetic one. It would be interesting to apply the proposed approach to a real system, such as the nuclear reactor discussed in (de Albornoz 1996). Furthermore, the approach should be compared to the PD–method to verify that it indeed outperforms the approaches that had been used traditionally for tackling the early warning problem.

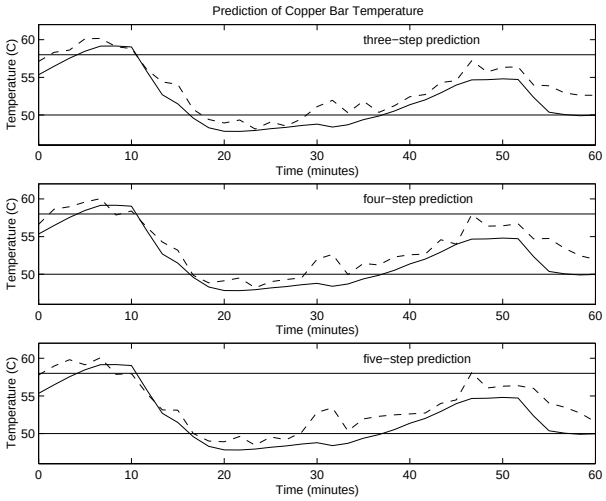

Figure A.7: Three–step to five–step predictions of copper bar temperature.

Appendix B

Signal Predictive Control

B.1 Introduction

In Appendix A, *smart sensors* were introduced as a means to improve system reliability, i.e., for reducing the probability of a complex engineering system to have to be shut down for safety considerations.

In the present appendix, smart sensors will be advocated as a tool to improve control performance. To this end, a new class of predictive control algorithms will be introduced, called *Signal Predictive Control (SPC)*.

The appendix explains the development of a strategy of predictive control that is based on the evaluation of predictions of the future behavior of the process to be controlled during a fixed time horizon in function of sequences of possible input/output behaviors within an umbrella of multiple predictions that are possible, given the available knowledge about the process and its feasible dynamics.

In a control process, two different types of input variables exist: *control variables* and *disturbances*. Only control variables can be manipulated by the controller. Although the disturbances influence the behavior of the system, the controller has no influence over them. The disturbances are divided into those that are *measurable* and those that are not.

Among the many control strategies that were devised by scientists and engineers over the past half century, two concepts have had the most profound impact on today's industrial control systems:

1. the *proportional, integral, and derivative (PID)* controller; and
2. the *model predictive control (MPC)* methodology.

The **PID controllers** and their simplified cousins: the PI, PD, and P controllers, all operate on the same principle. A measurable system output,

y, is fed back, and is compared with the desired value that it should attain, r, i.e., the so–called set value, producing an error signal, e:

$$e = r - y \tag{B.1}$$

The error signal is then amplified by a gain factor, k_P, to produce the proportional part of the controller. The error signal may also be integrated and amplified by another gain factor, k_I, to produce the integral part of the controller. It may finally be differentiated, and amplified with a third gain factor, k_D, to form the derivative portion of the controller. In this way, the control signal, u, is computed from the error signal, e, using the formula:

$$u(s) = \left(k_P + \frac{k_I}{s} + k_D \cdot s \right) \cdot e(s) \tag{B.2}$$

PID controllers have been widely surveyed in the open literature (Ogata 1970; Dorf 1980; Kuo 1991).

In practice, since numerical differentiation is considered harmful, it may be preferred to feed the derivative of the output, $\dot{y}(t)$, back also, and use the control law:

$$u(t) = k_P \cdot e(t) + k_I \cdot \int_0^t e(\tau) \cdot d\tau - k_D \cdot \frac{dy(t)}{dt} \tag{B.3}$$

This is equivalent to Eq.(B.2), because:

$$\frac{de}{dt} = \frac{dr}{dt} - \frac{dy}{dt} = -\frac{dy}{dt} \tag{B.4}$$

as the set value, $r(t)$, is constant.

The proportional part of the controller is responsible for bringing the output to the vicinity of its desired value; the integral part is responsible for the elimination of steady–state errors, i.e., for making the true output exactly equal to the desired output in steady–state; and the derivative portion is responsible for speed, i.e., for reducing the time that the controller needs to compensate for the influence of disturbances on the controlled output. A *PI controller* is a PID controller with $k_D = 0$, a *PD controller* is a PID controller with $k_I = 0$, and a *P controller* is a PID controller with both $k_D = 0$ and $k_I = 0$.

Figure B.1 shows the conceptual architecture of a PID controller. Its usual realization, that includes feedback of $\dot{y}(t)$, is not shown here, because this is an implementational detail. It does not change the functionality of the architecture.

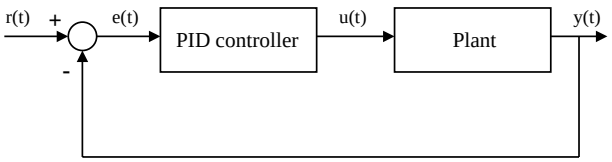

Figure B.1: PID control architecture

The PID controller owes its reputation to its *simplicity* and its *robustness*. The architecture can be applied to a large variety of different plants, including highly non–linear ones, and modifications of the architecture exist that can be used if either multiple control inputs influence the same system output, or if multiple outputs are being controlled through the same control input, as shown in Figure B.2.

Figure B.2(a) shows a plant with a single input and two outputs that are controlled by two separate PID controllers. The control signals produced by the two controllers are superposed at the single input of the plant. Figure B.2(b) shows a plant with two separate inputs that both influence the same output. A single error signal is computed that is fed to two separate PID controllers that control the two control inputs of the plant independently.

For many practical industrial engineering problems, the PID controller is all that it takes to satisfy the system requirements.

The **predictive control methodology** was introduced in 1974 in a doctoral thesis by Martín Sánchez (1974), and further developed by Martín Sánchez (1976), de Keyser (1991), Richalet *et al.* (1978), and Martín Sánchez and Rodellar (1996).

Today, there exist many different dialects of the basic predictive control

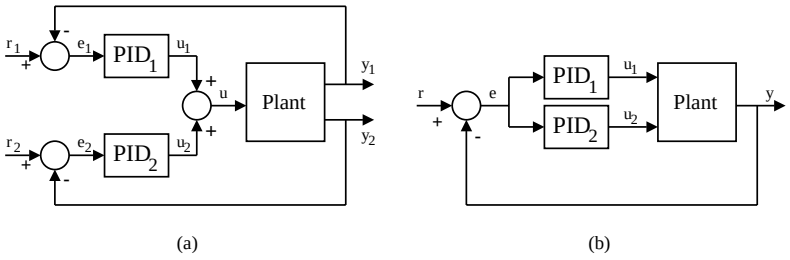

Figure B.2: Multivariable PID control architectures: (a) SIMO plant; (b) MISO plant

concept. The following architectures have been used to design predictive controllers:

- Model–based Predictive Control (MPC) (Clarke 1994; Qin 1998).
- Generalized Predictive Control (GPC) (Clarke *et al.* 1987a; Clarke *et al.* 1987b),
- Dynamic Matrix Control (DMC) (Cutler and Ramaker 1980),
- Extended Prediction Self–adaptive Control (de Keyser and van Cauwenberghe 1985),
- Predictive Functional Control (PFC) (Richalet *et al.* 1987),
- Extended Horizon Adaptive Control (EHAC) (Ydstie 1984)
- Unified Predictive Control (UPC) (Soeterboek *et al.* 1990a; Soeterboek *et al.* 1990b; Soeterboek 1992).

Most commonly, predictive controllers are designed in discrete time, but it is also possible to design them in continuous time (Gawthrop *et al.* 1996).

Figure B.3 shows the basic predictive control architecture. Predictive controllers are essentially *adaptive controllers*. A *controller design* algorithm adjusts the parameters of the controller using information provided by a model of the plant.

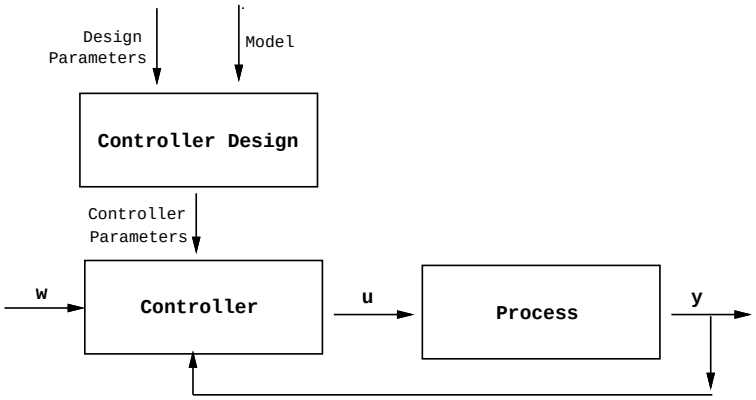

Figure B.3: Model–based predictive control architecture

The architecture makes use of a model of the plant in order to predict the future behavior of the plant given the current control strategy. A controller design algorithm makes use of that information in modifying the actual control strategy such that the model behavior approaches the desired plant behavior.

The different dialects of predictive controllers advocated in the open literature vary in the ways they implement and use the model in order to decide on an optimal control strategy.

In this appendix, a different type of predictive control architecture will be presented, an architecture that does not make use of a true plant model. It only models the output signal from observations of its own past using FIR. In this sense, the new architecture, which has been coined *Signal Predictive Control (SPC)*, is much simpler and more robust than most of the previously introduced predictive control strategies.

B.2 The Signal Predictive Control Architecture

Starting from the basic PID control architecture of Figure B.1, the modified PI controller architecture of Figure B.4 can be obtained. It is based on the

trivial equality:

$$y(t) = k \cdot y(t) + (1.0 - k) \cdot y(t) \tag{B.5}$$

which is evidently correct for any value of k.

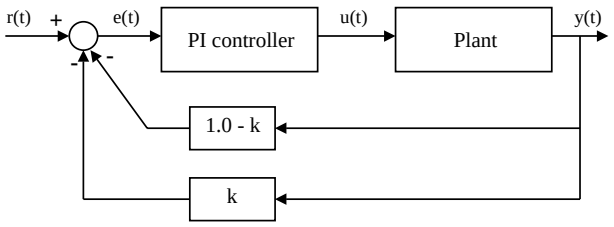

Figure B.4: A PI controller with redundant feedback loops

Using the control architecture of Figure B.4, the basic architecture of the new class of *signal predictive controllers* can be derived as shown in Figure B.5.

The box shown as *FIR* in Figure B.5 is a *smart sensor* with look–ahead capability. It predicts the value of the output $y(t)$ some Δt time units into the future.

The predictive component of the SPC architecture corresponds essentially to the introduction of a derivative term into the PI controller. However, it may have two important advantages over the PID controller:

1. Due to its non–linear nature, the FIR predictor may be able to better exploit the characteristics of a non–linear plant than the PID controller.

2. If the derivative of the output to be controlled is not a physical measurable variable, the performance of the PID controller is

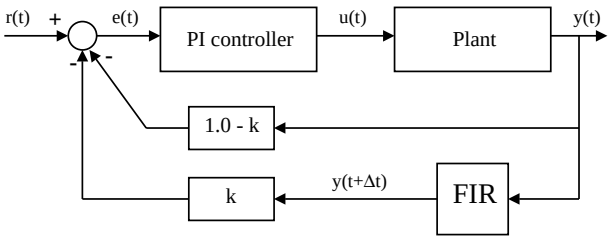

Figure B.5: Basic signal predictive control architecture

significantly reduced. In contrast, FIR does not require measuring a derivative signal.

If the derivative of the output is not a signal that is available through measurements, the derivative term of the PID controller:

$$u_D(t) = -k_D \cdot \frac{dy}{dt} \tag{B.6}$$

in accordance with Eq.(B.3), must be approximated. This can either be done in the frequency domain:

$$u_D(s) \approx -k_D \cdot \frac{s + 0.1}{s + 1} \cdot y(s) \tag{B.7}$$

or in the time domain:

$$u_D(t) \approx -k_D \cdot \frac{y(t) - y(t - \Delta t)}{\Delta t} \tag{B.8}$$

A better approximation in the time domain would be:

$$u_D(t) \approx -k_D \cdot \frac{y(t + \Delta t) - y(t)}{\Delta t} \tag{B.9}$$

Eq(B.9) shows the close relationship with the SPC architecture. Since $y(t + \Delta t)$ cannot be known precisely at time t, it would be possible to use FIR to estimate $y(t + \Delta t)$, then plug this approximation into Eq(B.9), in order to compute a better approximation of the derivative term. However, the estimate of $y(t + \Delta t)$ can also be used directly, as proposed in the SPC architecture.

The SPC architecture is *non–intrusive*, as it approaches smoothly the behavior of the PI controller for either $k \to 0$ or $\Delta t \to 0$. An adaptive SPC method would start out with small values of k and Δt, and then gradually increase the values of these two parameters, until the control performance is optimal.

The estimate of $y(t + \Delta t)$ can be improved by providing FIR also with the control input u as a second input. This modified SPC architecture is shown in Figure B.6.

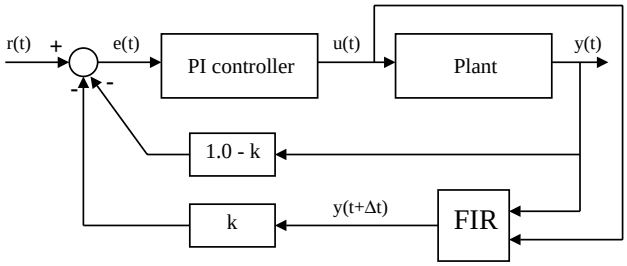

Figure B.6: Enhanced signal predictive control architecture

In the enhanced SPC architecture, FIR essentially models the plant, just like any other predictive controller would. However, the result of this model is not being used to modify the control strategy. Instead, it is used to modify the error signal that drives the controller.

However, since the aim of this dissertation is related to time–series

analysis, the discussion of the capabilities of the enhanced SPC architecture will be left to future research. This appendix only deals with the basic SPC architecture.

B.3 Application: The Copper Bar

A PI controller, as shown in Figure B.4, was built around the copper bar that had been introduced in Appendix A of this dissertation. The set value of the temperature is 50°C. The PI controller was optimized as to minimize the deviation of the measured bar temperature from its set value, once steady–state has been reached.

Whereas it is customary in the control literature to measure the effectiveness of a controller in terms of the integrated square error:

$$PI = \int_{t_0}^{t_f} err^2(\tau) \cdot d\tau \stackrel{!}{=} \min \tag{B.10}$$

in this dissertation, the performance index, PI, shall be defined as the *total error*:

$$PI = err_{\text{tot}} \stackrel{!}{=} \min \tag{B.11}$$

where err_{tot} is defined as presented in Eq(3.44).

The two parameters of the PI controller, k_P and k_I, were optimized as to make the total error minimal during steady–state.

The controlled temperature of the copper bar is shown in Figure B.7.

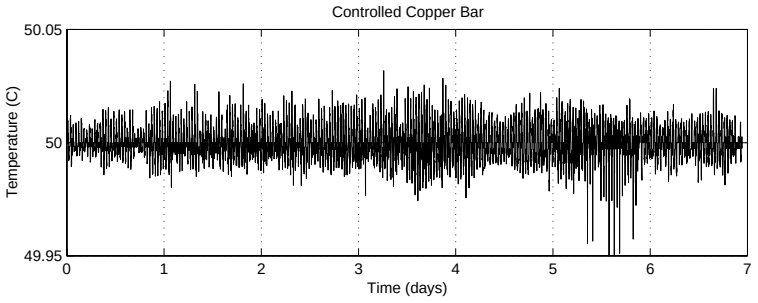

Figure B.7: Controlled copper bar temperature

The time series of Figure B.7 of the controlled copper bar temperature has been coined Series C. Series C can be characterized as follows:

Series C is similar to Series U, except that it should be characterized as stationary, rather than non–stationary. Series C looks treacherously like

Table B.1: Classification of Time Series C

natural		synthetic	C
stationary	C	non–stationary	
time invariant	C	time varying	
low dimensional		stochastic	C
clean		noisy	C
short		long	C
dormant		active	C
documented	C	blind	
linear		non–linear	C
scalar	C	vector	
single recording	C	multiple recordings	
continuous	C	discrete	

white noise, i.e., it will be difficult to extract any useful information out of this series.

Using the first 5000 data records, a FIR model was constructed. It has the structure:

$$\begin{array}{ll} time & u_1 \\ t-48\delta t & \left(\begin{array}{r} -1 \\ 0 \\ -2 \\ 0 \\ -3 \\ 0 \\ -4 \\ 0 \\ +1 \end{array}\right. \\ \cdots & \\ t-24\delta t & \\ \cdots & \\ t-9\delta t & \\ \cdots & \\ t-2\delta t & \\ t-\delta t & \\ t & \end{array} \tag{B.12}$$

Interestingly enough, FIR did not consider the value at time $t-\delta t$ useful for predicting the value at time t. This by itself should make us suspicious about the quality of this data stream.

Since the output of the FIR model is used to drive the controller and plant models that are simulated in ACSL (MGA 1998), also the FIR simulation had to be performed in ACSL. This was accomplished using the FIR run–time kernel of ACSL that had been developed in (Cellier *et al.* 1992).

Figure B.8 shows the results of a simulation across 27.77 hours, corresponding to 1000 samples. The top graph of Figure B.8 shows the

simulation using the optimized PI controller; the center portion of Figure B.8 shows the simulation using an optimized PID controller, where all three parameters, k_P, k_I, and k_D, were re–optimized to minimize the chosen performance index; and the bottom part of Figure B.8 shows the simulation using the signal predictive controller with a look–ahead of one step at a time ($\Delta t = 1$ step $= 1.66$ minutes), whereby the remaining three parameters, k_P, k_I, and k, were re–optimized for minimization of the performance index.

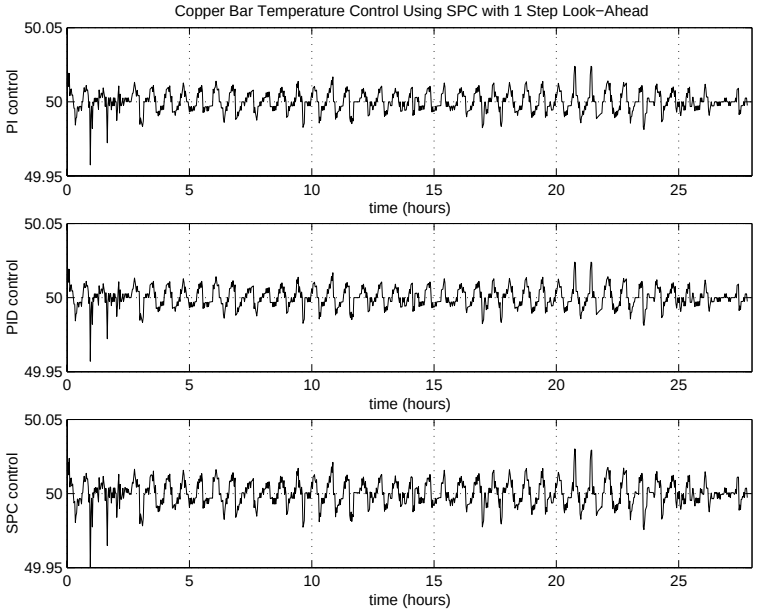

Figure B.8: Comparison of PI, PID, and SPC architectures for copper bar temperature control

The three curves look almost indistinguishable by naked eye. One might be inclined to believe that the PI controller is best, whereas the SPC is worst. Figure B.9 compares the SPC using a double–step look–ahead, a triple–step look–ahead, and a quadruple–step look–ahead.

Again, not much is accomplished when looking at the simulation results by naked eye. Table B.2 summarizes the simulation results in a tabular form.

Although not visible by the naked eye, the performance of the SPC architecture was indeed slightly better than that of the PID controller (at least, in a strictly numerical sense), which performed a little better than the PI controller. The optimal number of steps of look–ahead for this system is two.

The numerical "improvement" obtained by the SPC approach certainly does not justify the effort for the example at hand. In practice, one should

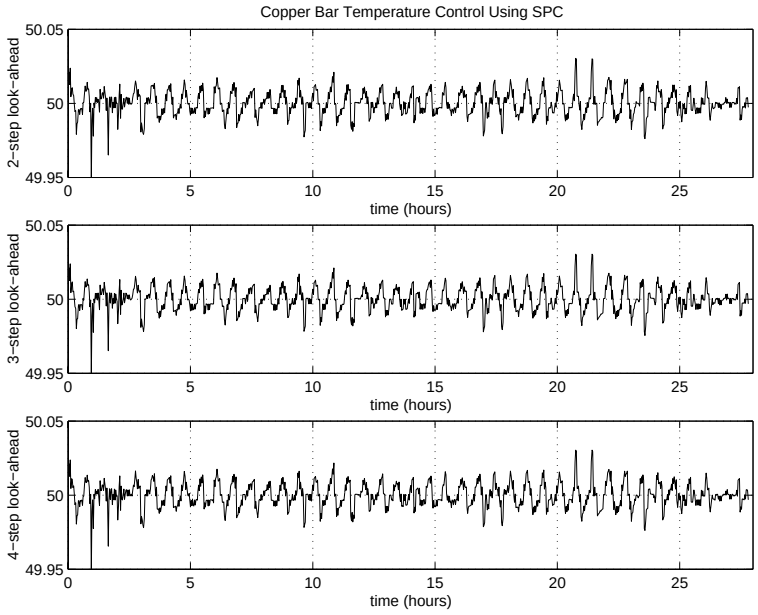

Figure B.9: Comparison of SPC architectures for copper bar with multiple steps look–ahead

Table B.2: Error of the Controller

PI Controller	PID Controller	SPC Architecture n - step Prediction			
		n = 1	n = 2	n = 3	n = 4
29.7422	29.7207	29.6362	29.5589	29.6068	29.6352

not even call this an "improvement" at all. Moreover, the PI controller satisfies the control performance criteria that can reasonably be expected of this system. It is much simpler, and therefore better.

The fact that also the PID controller did not bring any significant improvement shows that the example was not well chosen. The SPC architecture is hypothesized to work well when applied to systems where the PID controller shows a significant improvements in performance over that of the PI controller. In such systems, it may be speculated that the SPC approach would outperform the PID controller.

B.4 Conclusion

A new predictive control architecture, called *Signal Predictive Control (SPC)*, was proposed that has a number of appealing properties:

1. The SPC architecture is non–intrusive, i.e., for small values of its two parameters, k and Δt, it performs essentially like the well–known and well–liked, highly–robust PI controller.

2. The SPC architecture is simple and logical. It does not call for additional measurement data, such as measured derivatives.

3. The SPC architecture can be generalized to multivariable systems as easily as the PID control architecture.

Only a single control system was analyzed to this day using the SPC architecture. The results of this analysis are somewhat disappointing, because the example was poorly chosen for the methodology.

Yet, the author is convinced that SPC has a bright and promising future, and will find a proud place among the family of predictive control approaches.

The results presented in the two appendices must certainly be called *preliminary* and *speculative*. A significant effort was already spent on developing the new monitorization and control architecture, and it would therefore have been a pity to leave the results obtained so far out of the dissertation altogether. Yet, in the light of the preliminary nature of these findings, it was felt that it would be more appropriate to present them in two appendices, rather than in the main body of the dissertation.

Bibliography

Aigües de Barcelona, S. (1985). Water demand data for the city of Barcelona.

Baggelaar, P. (1992). Voorspelling van het dagverbruik in het voorzieningsgebied van de berenplaat. *H_2O 25*(3), 71–74.

Box, G. E. P. and F. M. Jenkins (1994). *Time Series Analysis: Forecasting and Control*. Englewood Cliffs, NJ: Prentice Hall.

Brockwell, P. J. and R. A. Davis (1991). *Time Series: Theory and Methods*. New York: Springer-Verlag.

Brockwell, P. J. and R. A. Davis (1996). *Introduction to Time Series and Forecasting*. New York: Springer-Verlag.

Burr, T. (1998). Comparison of fuzzy forecaster to a statistically motivated forecaster. *IEEE Transactions on Systems, Man & Cybernetics, Part A 28*(1), 121–7.

Casdagli, M. and S. Eubank (Eds.) (1992). *Nonlinear Modeling and Forecasting*. Addison-Wesley.

Casdagli, M. C. (1991). Chaos and deterministic versus stochastic nonlinear modeling. *Journal Roy. Stat. Soc. B 54*, 303–328.

Cellier, F. (1987). Qualitative simulation of technical systems using the general systems problem solving framework. *International Journal of General Systems 13*(4), 333–344.

Cellier, F. and J. López (1995). Causal inductive reasoning: A new paradigm for data-driven qualitative simulation of continuous-time dynamical systems. *Systems Analysis Modelling Simulation 18*(1), 27–43.

Cellier, F., J. López, A. Nebot, and G. Cembrano (1996). Means for estimating the forecasting error in fuzzy inductive reasoning. In *Procedings of the ESM'96 Ontl. Conf on Qualitative Information Fuzzy*

Systems and Neural Networks in Simulation, Budapest, Hungary, pp. 654–660.

Cellier, F., J. López, A. Nebot, and G. Cembrano (1998). Confidence measure in fuzzy inductive reasoning. *International Journal of General Systems*, in print.

Cellier, F., A. Nebot, F. Mugica, and A. d. Albornoz (1992). Combined qualitative/quantitative simulation models of continuous–time processes using fuzzy inductive reasoning techniques. In *Proceedings SICICA'92, IFAC Symposium on Intelligent Components and Instruments for Control Applications*, Malaga, Spain, pp. 589–593.

Cellier, F., A. Nebot, F. Mugica, and A. d. Albornoz (1996). Combined qualitative/quantitative simulation models of contnuous–time processes using fuzzy inductive reasoning techniques. *International Journal of General Systems 24*(1–2), 95–116.

Cellier, F. and D. Yandell (1987). Saps–II: A new implementation of the systems approach problem solver. *International Journal of General Systems 13*(4), 307–322.

Cellier, F. E. (1991). *Continuous System Modeling*. New York: Springer Verlag.

Chabot, I. (1998). 5–day weather forecast, senior project, University of Arizona, Tucson.

Chang, P.-T. (1996). Fuzzy seasonality forecasting. *Fuzzy Sets and Systems 90*(1), 1–10.

Chatfield, C. (1989). *The Analysis of Time Series*. London: Chapman and Hall.

Chen, S.-M. (1996). Forecasting enrolment based on fuzzy time series. *Fuzzy Sets and Systems 81*(3), 311–19.

Clarke, D. W. (1994). *Advances in Model-Based Predictive Control*. Oxford University Press.

Clarke, D. W., C. Mohtadi, and P. S. Tuffs (1987a). Generalized predictive control Part I. The basic algorithm. *Automatica 23*(2), 137–148.

Clarke, D. W., C. Mohtadi, and P. S. Tuffs (1987b). Generalized predictive control Part II. Extensions and interpretations. *Automatica 23*(2), 149–160.

Connor, J., L. E. Atlas, and D. R. Martin (1992). Recurrent network and NARMA modeling. In *Advances in Neural Information Processing Systems*, Volume 4, pp. 301–308.

Cottrell, M., B. Girard, Y. Girard, M. Mangeas, and C. Muller (1995). Neural modeling for time series: a statistical stepwise method for weight elimination. *IEEE Transaction on Neural Networks 6*, 1355–1364.

Cutler, C. R. and B. L. Ramaker (1980). Dynamic matrix control - a computer control algorithm. In *Proceedings Jacc*, San Francisco, U.S.A.

de Albornoz, A. (1996). *Inductive Reasoning and Reconstruction Analysis: Two Complementary Tools for Qualitative Fault Monitoring of Large–Scale Systems*. Ph. D. thesis, Universitat Politècnica de Catalunya, Barcelona.

de Keyser, R. M. C. (1991). Principles of model based predictive control. In *Proceedings First European Control Conference*, Grenoble, France, pp. 1753–1758.

de Keyser, R. M. C. and A. R. van Cauwenberghe (1985). Extended prediction self–adaptive control. In *Proceedings 7th IFAC Symposium on Identification and System Parameter Estimation*, York, UK, pp. 1255–1260.

Delgado, A. (1998). *Redes de Neuronas: Aportaciones Teóricas y Prácticas a su Diseño e Implementación*. Ph. D. thesis, Universitat Politècnica de Catalunya, Barcelona.

Dorf, R. C. (1980). *Modern Control Systems*. Reading, Mass: Addison–Wesley.

Dubois, D. and H. Pradé (1980). *Fuzzy Sets and Systems, Theory and Applications*. New York: Academic Press.

Dynasim (1996). *Dymola: Dynamic Modeling Laboratory – User's Manual*. Lund, Sweden: Dynasim S.A.

Europoort, W. (1986). Water demand data for the city of Rotterdam.

Gawthrop, P. J., R. W. Jones, and D. G. Sbarbaro (1996). Emulator-based control and internal model control: Complementary approaches to robust control design. *Automatica 32*(8), 1223–1227.

Gershenfeld, N. A. and A. S. Weigend (1994). The future of time series: Learning and understanding. In A. S. Weigend and N. A. Gershenfeld (Eds.), *Time Series Prediction: Forecasting the Future and Understanding the Past*, Reading, MA, pp. 1–70. Addison-Wesley.

Ghoshray, S. (1996). Currency exchange rate prediction technique by fuzzy inferencing on the chaotic nature of time series data. *International*

Journal of Uncertainty, Fuzziness and Knowledge Based Systems 4(5), 431–48.

Golob, R., T. Stokelj, and D. Grgic (1998). Neural-network-based water inflow forecasting. *Control Engineering Practice 6*(5).

Griñó, R. (1992). Neural network for univariate time series forecasting and their application to water demand prediction. *Neural Network World*, 437–450.

Guo, L., L. Ljung, and G.-J. Wang (1997). Necessary and sufficient condition for stability of lms. *IEEE Transactions On Automatic Control 42*(6), 761–770.

Haber, R. and H. Unbehauen (1990). Structure identification of nonlinear dynamic systems a survey on input/output approaches. *Automatica 4*(26), 651 –677.

Ishikawa, M. and T. Moriyama (1996). Prediction of time series by a structural learning of neural networks. *Fuzzy Sets and Systems 82*(2), 167–76.

Ivanova, T. O., V. V. Mottle, and I. B. Muchnik (1994). Estimation of the parameters of hidden Markov models of noise-like signals with abruptly changing probabilistic properties. *Automation and Remote Control 55*, 1299–1315,1428–1445.

Jang, J.-S. R. (1997). *Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence*. Upper Saddle River, NJ: Prentice Hall.

Karr, C. L. and E. J. Gentry (1993). Fuzzy control of ph using genetic algorithms. *IEEE Trans. Fuzzy Sytems 1*(1), 46–53.

Kim, D. and C. Kim (1997). Forecasting time series with genetic fuzzy predictor ensemble. *IEEE Transactions on Fuzzy systems 5*(4), 523–535.

Klir, G. J. (1985). *Architecture of Systems Problem Solving*. New York: Plenum Press.

Klir, G. J. and T. Folger (1988). *Fuzzy Sets, Uncertainty and Information*. Englewood Cliffs, NJ: Prentice Hall.

Klir, G. J. and B. Yuan (1995). *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Upper Saddle River, NJ: Prentice Hall PTR.

Korn, G. A. (1995). *Neural Networks and Fuzzy-Logic Control on Personal Computers and Workstations*. Cambridge, MA: MIT Press.

Kosko, B. (1991). *Neural Networks for Signal Processing*. Englewood Cliffs, NJ: Prentice Hall.

Kosko, B. (1992). *Neural Networks and Fuzzy Systems - A Dynamical Systems Approach to Machine Intelligence*. Englewood Cliffs, NJ: Prentice Hall.

Kuipers, B. and A. Farquhar (1987). *QSim: A Tool for Qualitative Simulation*. Artificial Intelligence Laboratory, The University of Texas, Austin.

Kuo, B. C. (1991). *Automatic Control Systems*. Englewood Cliffs, NJ: Prentice Hall.

Law, A. and D. Kelton (1990). *Simulation Modeling and Analysis*. Englewood Cliffs, NJ: Prentice Hall.

Lee, C. C. (1990). Fuzzy logic in control systems: Fuzzy logic controller. *IEEE Trans. Syst. Man. Cybern 22*(2), 403–435.

Li, D. and F. E. Cellier (1990). Fuzzy measures in inductive reasoning. In *Proceedings of the 1990 Winter Simulation Conference*, New Orleans, LA, pp. 527–538.

Ljung, L. (1987). *System Identification: Theory for the User*. Information and System Sciences Series. Englewood Cliffs, NJ: Prentice Hall.

Ljung, L. (1992). *Stochastic Approximation and Optimization of Random Systems*. Basel; Boston: Birkhäuser Verlag.

López, J., G. Cembrano, and F. E. Cellier (1996). Time series prediction using fuzzy inductive reasoning. In *Proceedings ESM'96: European Simulation Multiconference*, Budapest, Hungary, pp. 765–770.

Makridakis, S., A. Andersen, R. Carbone, R. Fildes, M. Hibon, R. Lewandowski, J. Newton, E. Parzen, and R. Winkler (1984). *The Forecasting Accuracy of Major Time Series Methods*. New York: John Wiley.

Makridakis, S. and M. Hibon (1979). Accuracy of forecasting: An empirical investigation. *J. Roy. Stat. Soc. A 142*, 97–145.

Martín Sánchez, J. M. (1974). *Contribution to model reference adaptive systems from hyperstability theory (in Spanish)*. Ph. D. thesis, Universitat Politècnica de Catalunya, Barcelona, Spain.

Martín Sánchez, J. M. (1976). *Adaptive predictive control system*. USA Patent No. 4,197,576.

Martín Sánchez, J. M. and J. Rodellar (1996). *Adaptive Predictive Control*. International Series in Systems and Control Engineering. Great Britain: Prentice Hall.

MathWorks (1997). *Getting Started with MATLAB; Using MATLAB (Version 5.1)*. Natick, MA: The MathWorks Inc.

MGA (1998). *ACSL: Advanced Continuous Simulation Language – Reference Manual*. Mitchell & Gauthier Assoc., Concord, Mass., USA.

Mitsch, W. and B. Marino (1999). *Ecological Engineering – Special Issue on Biosphere 2 13*(1-4).

Moorthy, M. (1999). Mixed structural and behavioral models for predicting the future behavior of some aspects of the macroeconomy, MS thesis, University of Arizona, Tucson.

Moorthy, M., F. E. Cellier, and J. T. LaFrance (1998). Predicting U.S. food demand in the 20th century: A new look at system dynamics. In *Proceedings of the SPIE Conference*, Volume 3369, Orlando, Florida, pp. 343–354.

Mugica, F. (1995). *Diseño Sistemático de Controladores Difusos Mediante Razonamiento Inductivo*. Ph. D. thesis, Universitat Politècnica de Catalunya, Barcelona.

Muller, C., M. Cottrell, M. Girard, B. Girard, and M. Mangeas (1994). A neural network tool for forecasting French electricity consumption. In *Proceedings of WCNN'94*, San Diego, California, USA.

Narendra, K. S. and S. M. Li (1995). Neural networks in control systems. In P. Smolensky, M. C. Mozer, and D. E. Rumelhart (Eds.), *Mathematical Perspective on Neural Networks*, Hillsdale, NJ. Lawrence Erlbaum Associates.

Narendra, K. S. and S. Mukhopadhyay (1995). Neural networks in dynamical systems. In P. Smolensky, M. C. Mozer, and D. E. Rumelhart (Eds.), *Mathematical Perspectives on Neural Networks*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Narendra, K. S. and K. Parthasarathy (1990). Identification and control of dynamical systems using neural networks. *IEEE Trans. on Neural Networks 1*, 4–27.

Nebot, A. (1994). *Qualitative Modeling and Simulation of Biomedical Systems using Fuzzy Inductive Reasoning*. Ph. D. thesis, Universitat Politècnica de Catalunya, Barcelona.

NeuralWare (1993). *NeuralWorks Professional II Plus*. NeuralWare Inc.

Ogata, K. (1970). *Modern Control Engineering*. Englewood Cliffs, NJ: Prentice Hall.

Oppenheim, A. V. and R. W. Schafer (1989). *Discrete-Time Signal Processing*. Englewood Cliffs, NJ: Prentice Hall.

Papadakis, S. E., J. B. Theocharis, S. J. Kiartzis, and A. G. Bakirtzis (1998). A novel approach to short–term load forecasting using fuzzy neural networks. *IEEE Transactions on Power Systems 13*(2), 480–489.

Priestley, M. (1981). *Spectral Analysis and Time Series*. London: Academic Press.

Qin, S. J. (1998). Control performance monitoring – a review and assessment. In *NSF/NIST Workshop on Measurement and Control. Also accepted by Comp. and Chem. Engng.*

Quevedo, J., G. Cembrano, A. Valls, and J. Serra (1988). Time series modelling of water demand — a study on short–term and long–term predictions. In B. Coulbeck and C. Orr (Eds.), *Computer Applications in Water Supply*, pp. 268–288. John Wiley.

Richalet, J., S. A. el Alta-Doss, C. Arber, H. Kuntze, A. Jacubasch, and W. Schill (1987). Predictive functional control. application to fast and accurate robots. In *Proceedings 10th IFAC World Congress*, Munich, Germany.

Richalet, J., A. Rault, J. L. Testud, and J. Papon (1978). Model predictive heuristic control: applications to industrial processes. *Automatica 14*(5), 413–428.

Sarjoughian, H. (1995). *Inductive Modeling of Discrete-Event Systems: A TMS-Based Non-Monotonic Reasoning Approach*. Ph. D. thesis, University of Arizona, Tucson.

Shannon, C. (1951). Prediction and entropy of printed English. *Bell System Technical Journal 30*, 50–64.

Soeterboek, A. R. M., H. B. Verbruggen, P. P. J. van den Bosch, and H. Butler (1990a). Adaptive predictive control – a unified approach. In *Proceedings Sixth Yale Workshop on Applications of Adaptive Systems Theory*, New Haven, U. S. A.

Soeterboek, A. R. M., H. B. Verbruggen, P. P. J. van den Bosch, and H. Butler (1990b). On the unification of predictive control algorithms. In *Proceedings 29th IEEE Conference on Decision and Control*, Honolulu, U. S. A.

Soeterboek, R. (1992). *Predictive Control: A Unified Approach.* International Series in Systems and Control Engineering. NJ,USA: Prentice Hall.

Stahl, G. (1996). Armchair missions to Mars: Using case-based reasoning and fuzzy logic to simulate a time series model of astronaut crews. *Knowledge–Based Systems 9*(7), 409–15.

Takagi, T. and M. Sugeno (1991). Nn–driven fuzzy reasoning. *International Journal of Approximate Reasoning 5*(3), 191–212.

Takens, F. (1981). Detecting strange attractors in turbulence. In D. A. Rand and L. S. Young (Eds.), *Dynamical Systems and Turbulence*, Volume 898 of *Lecture Notes in Mathematics*, pp. 366–381. Springer.

Tanaka, K., M. Sano, and H. Watanabe (1995). Modeling and control of carbon monoxide concentration using a neuro–fuzzy technique. *IEEE Trans. Fuzzy Sytems 3*(3), 271–279.

Tong, H. (1990). *Non-linear Time Series: A Dynamical System Approach.* Oxford University Press.

Uyttenhove, H. (1978). *Computer–Aided Systems Modeling: An Assemblage of Methodological Tools for Systems Problem Solving.* Ph. D. thesis, School of Advancd Technology, University of New York, SUNY–Binghamton, USA.

Volterra, V. (1959). *Theory of Functionals and of Integral and Integro-Differential Equations.* New York: Dover.

Wan, E. A. (1994). Times series prediction using a connectionist network with internal delay lines. In A. S. Weigend and N. A. Gershenfeld (Eds.), *Time Series Prediction: Forecasting the Future and Understanding the Past*, Reading, MA, pp. 195–217. Addison-Wesley.

Wang, L. and R. Langari (1995). Building Sugeno–type models using fuzzy discretization and orthogonal parameter estimation techniques. *IEEE Trans. Fuzzy Sytems 3*(4), 454–458.

Weigend, A. S. and N. Gershenfeld (Eds.) (1994). *Time Series Prediction: Forecasting the Future and Understanding the Past.* Reading, MA: Addison-Wesley.

Weigend, A. S., B. A. Huberman, and D. E. Rumelhart (1990). Predicting the future: A connectionist approach. *International Journal of Neural Systems 1*, 193–209.

Weigend, A. S. and M. Mangeas (1995). Avoiding overfitting by locally matching the noise level of the data. In *World Congress on Neural Networks (WCNN'95)*, pp. II–1–9.

Weigend, A. S., M. Mangeas, and A. N. Srivastava (1995). Nonlinear gated experts for time series: Discovering regimes and avoiding overfitting. *International Journal of Neural Systems 6*, 373–399.

Weigend, A. S. and D. A. Nix (1994). Predictions with confidence intervals (local error bars). In *Proceedings of the International Conference on Neural Information Processing (ICONIP'94)*, Seoul, Korea, pp. 1207–1212.

White, D. A. and D. A. Sofge (Eds.) (1992). *Handbook of Intelligent Control*. Van Nostrand Reinhold.

Ydstie, B. E. (1984). Extended horizon adaptive control. In *Proceedings 9th IFAC World Congress*, Budapest, Hungary.

Yule, G. (1927). On a method of investigating periodicity in disturbed series with special reference to Wolfer's sunspot numbers. *Phil. Trans. Roy. Soc. London A 226*, 267–298.

Zhang, P. and R. Li (1996). Fuzzy identification through fuzzy clustering techniques and kalman filter method. *Control Theory & Applications 13*(5), 639–43.

Zimmermann, H. G. and A. S. Weigend (1995). New ways to use time in time series modeling. Technical Report CU-CS-796-95, University of Colorado at Boulder, Computer Science Department.