Alois Geyer and Andreas Geyer-Schulz and Alfred Taudes

A Fuzzy time-series analyzer

A FUZZY TIME-SERIES ANALYZER

Alois Geyer, Andreas Geyer-Schulz, Alfred Taudes
Vienna University of Economics and Business Administration
Augasse 2-6, A-1090 Vienna

February 22, 1988

1 Motivation

This paper deals with a fuzzy set based expert system for analyzing time-series with ARIMA-models. The motivation to undertake this project arose out of a number of experiences made during the practical application of mathematical models, a review of the current state of the art of Operations Research-software and previous attempts in literature to represent and process model-building knowledge.

Productivity in mathematical modeling is low: reality is complex, and it usually takes a lot of trials to find a satisfactory mathematical description of the phenomenon under consideration. Due to this complexity, modeling has to be done by (costly) specialists, who are required to speak three "languages" - the language of mathematics in which the model is originally described, a programming language or an input-language to a standard package which is needed to solve the particular case, and the language of the user who is ignorant of these "internal representations", presents his problem in "user-terms" and also needs the relevant features of the model depicted via e.g. graphic means. Another practical problem associated with mathematical models is the "model-management problem". Modeling is not a once-and-for-all thing: it is necessary to update models, provide fast simulations or predictions and to perform data-manipulations. Current standard application packages (optimization packages, statistics packages, etc.) support only one part of the application-process - computing - but do not cover important tasks like data-management, model-structure selection, and training of the users.

2 Systems Architecture

So why not solve these problems with expert systems technology by enhancing numerical software by "modeling experts", i.e. software that covers the model-building process by assisting the model-builder or even by automatically generating a model? Decision support systems that allow mathematical modeling by providing theoretical guidance, computer aided teaching-programs which allow the student to develop his modeling skills under proper assistance, and systems that automate the routine-parts of the analysis could become a reality. But how should such a model-selection and model-management system work?

When consulting the literature on mathematical modeling, in general one finds that while the models themselves might be well-structured, the process of specification and application of a model to a given problem is dominated by ill-structured, often subjective heuristics. Consider, for example, the problem of identifying a proper ARIMA-model for a given time-series: two "experts", [Granger, Newbold], write "... the stage that causes the most difficulties in practical attempts at time series model building is identification. Here one is required to choose from a wide class of models a single process that might adequately describe a given time-series. While some objective criteria are available on which a rational choice can be based, it remains the case that there does not exist a clearly defined procedure leading in any given situation to a unique identification. Rather, it is necessary to excercise a good deal of judgement at this stage. "... "Two particular problems in model identification are worth mentioning. First, it is at times extremely difficult from a given set of data to identify a particular model from the general class in which one has a great deal of confidence. Up to a point, one finds this happening less frequently the more experience one has in using the techniques ..." and "... It is sometimes the case that multiple identifications are thought possible; that is, the identification procedure might suggest two or more models from the general class that could well represent a particular set of data ...".

Thus, a model-building expert, be it a human one or a program, has to act according to a heuristic. He, therefore, takes a subjective risk that is not reflected in the statistical uncertainty assigned to the output of the model and that only holds under the structural assumptions given. Thus its neglectance can be dangerous as it leaves the user of the model without a warning that structures other than the one chosen by the model-selection program might be valid descriptions for his problem too. Forecasting packages including the specification part known to the authors have thus dealt with this problem by giving "points" to specific structures via tables inside the system [Eckhardt] or by introducing "critical values" for statistics in a more or less arbitrary way.

The approach taken here is the modeling of the selection process explicitly via a rule base describing the expert's theoretical knowledge and modeling strategy: it can be easily understood, altered and extended by the user, and it also can be enhanced to infer problem-specific rules from a user-dialog. In the case of time-series analysis these rules are a set of inexact, verbal statements like "the autocorrelation function of an autoregressive process of order p tails off, its partial autocorrelation function has a cutoff after lag p." [Box, Jenkins]. They contain references to the particular properties of the time-series, (in the example given the order p of the autoregressive process), the combination of statistics used for their inference, (here the autocorrelation and the partial autocorrelation function), and the appropriate next step in the model-building procedure, i.e. in our example the estimation of an AR(p)-process.

A formal basis for such an inexact rule-based system is fuzzy set theory [Zadeh]. The basic premise of this extension of traditional set theory in our context is that the class of ARIMA-models constitutes a fuzzy set: it is proposed that a particular time-series is related to an ARIMA-model by a membership function that instead of being discretely restricted to 0 and 1 takes values in the real interval [0,1]. This function maps the observed time-series to the model according to the degree of plausibility that the observations are generated by this particular stochastic process. Thus several models can be entertained simultaneously. However, not only the models are treated as fuzzy sets, but also particalur relevant properties of the time-series like trend, seasonality or order.

The process of identification starts with the calculation of the relevant statistics which are subsequently transformed into fuzzy sets via pattern-matching routines. Then the higher-level properties are obtained via fuzzy set-operations on the lower-level results (for the tree of concepts used see 1). The presence of certain properties in the time-series can trigger an action (e.g. taking first differences in the case of a trend), thus we introduce a stack that contains the states a particular time-series is taking in the course of its analysis. The link between rule-base and fuzzy sets is accomplished via "natural language computations", i.e. two mappings, one transforming verbally given intermediate concepts into their internal representation as fuzzy sets and the other one mapping the fuzzy set resulting from an inference into a verbal statement. We implement this procedure using a frame-system that allows a separation of the structural part (e.g. the statistics to be used) from the type of logic chosen to combine the various concepts. Note that the program is organized in a completely modular way: in order to include a new statistic in the analysis it is only necessary to provide the formula, a transformation to a fuzzy set and a corresponding reference in the fuzzy rule-base.

Table 1: Concept Tree

High-Level Property	Low-Level Properties
Heteroscedascity	test significant
	trigger if found: apply appropriate transformation
Trend	AR(l) process with a parameter close to 1.0 sign test significant slowly decaying autocorrelations a peak at zero-frequency in the estimated spectrum
	trigger if found: differencing
Season	a peak in the estimated spectrum at a frequency different from 0 autocorrelations have a peak at corresponding lag
	trigger if found: seasonal differencing
Order	corresponding autocorrelation-pattern corresponding partial autocorrelation-pattern corresponding vectorcorrelation-pattern
	trigger: parameter-estimation
Proper Model	cumulative periodogram test ok AIC minimum SBC minimum Ljung-Box test ok
	trigger if ok: finish else: modify model or take another model found plausible and perform parameter-estimation

When implementing such a system, one has to integrate symbolic and numerical computing. On one hand the numerical routines exist and are well-tested, however, writing an inference mechanism for knowledge processing in the procedural languages used for numerical computing causes a prohibitively large effort. The popular programming languages for artificial intelligence applications, LISP and PROLOG, on the other hand, do not support exact numerical computing (e.g. the interpreters do not have double precision features, built-in numerical functions are lacking, etc.). Thus a common approach is to enhance the input-side of the user-interface of existing packages by providing assistance when selecting a method via knowledge-based questioning of the user and/or to enlarge the presentation of their results by providing an "expert"-interpretation of the numerical output. However, adding knowledge "in front" of a numerical package does not account for a multi-step identification procedure involving trial and error. Interpreting statistics after a numerical run is a "tabula rasa" strategy, whereby at first all kinds of calculations are performed and analyzed later. A combined approach induces a considerable processing overhead: the output of the numerical package has to be read and lexically scanned by a knowledge part added "at the rear" and a "model-chooser" added "in front" has to write an input-routine to the numerical program.

Our approach is to integrate both numerical and symbolic computing into one package, making use of the features of APL2. APL2 provides generalized arrays, i.e. arrays that can have arrays as their elements, and user-defined operators, i.e. functions, that can take functions as their arguments. These enhancements make this array-oriented functional programming language suitable for applications involving numerical and symbolic computing. For example, fuzzy sets are stored as generalized arrays containing a vector describing the base-set and a vector containing the values of the membership function. The stack containing the states of the time-series is implemented via generalized arrays, providing a convenient explanation of the system's functioning.

3 Applications

The system is currently subjected to a large number of tests in order to check its ability to correctly identifiy and forecast various simulated time-series and well-known data from literature. It is planned to use it as a tool to support teaching of time-series analysis and forecasting at the Vienna University of Economics and Business Administration. Furthermore it will serve as a basis of research that aims at checking the utility of forecasting methodologies from an empirical point of view by performing a sufficiently large number of automatic analyses and testing the improvements in comparison with other approaches.

References

[Box, Jenkins] Box G.E., and G.M. Jenkins (1976), *Time-Series Analysis Forecasting and Control*, Holden-Day, San Francisco

[Eckhardt] Eckhardt, Th. (1980), "Eine Prognose-Methodenbank fuer Kleinrechner", J. Schwarze, *Angewandte Prognoseverfahren*, Neue Wirtschaftsbriefe, Berlin, 115-133

[Granger, Newbold] Granger C.W.J., and P. Newbold (1977), *Forecasting Economic Time-Series*, Academic Press, New York

[Zadeh] Zadeh L.A. (1965), "Fuzzy Sets", *Information and Control*, 8, 338-353