Full citation: MacDonell, S.G., Gray, A.R., & Calvert, J.M. (1999) FUZZYMANAGER: a teaching and introductory environment for fuzzy logic and fuzzy clustering, in Proceedings of the ICONIP'99/ANZIIS'99/ANNES'99/ACNN'99 International Workshop on Future Directions for Intelligent Systems and Information Sciences. Dunedin, New Zealand, University of Otago, pp.197-202.

FUZZYMANAGER: A Teaching and Introductory Environment for Fuzzy Logic and Fuzzy Clustering

Stephen G. MacDonell, Andrew R. Gray, and James M. Calvert
Department of Information Science
University of Otago, PO Box 56, Dunedin, New Zealand
+64 3 4798135 (ph.) +64 3 4798311 (fax), stevemac@infoscience.otago.ac.nz

Abstract

Using fuzzy logic, and associated techniques such as fuzzy clustering, in a teaching environment necessitates the availability of introductory and pedagogically appropriate tools. In a similar manner, introductory level tools may be necessary for practical applications where users are non-specialists in fuzzy theory, as is often the case. For these two scenarios, and many others, the tools that support the use of the modeling technique must satisfy sets of requirements concerning the interface, functionality, and documentation. Examples of these requirements can include the program's ability to guide the user, without undue restrictions, through the necessary modeling procedures (as in a wizard interface) and explaining (both textually and graphically) the operation of the inference process. After outlining some of the desirable attributes for such tools that are intended to be used by these fuzzy novices, this paper describes the collection of tools collectively known as FUZZYMANAGER. Despite the name, which reflects its project management origins, the system is targeted towards any group of users without a particularly comprehensive or deep knowledge of fuzzy logic, who want an intuitive and graphical approach to fuzzy logic model building. As well as implementing standard inference options and methods, a clustering based algorithm for automatically deriving fuzzy systems from data (either membership functions, rules, or both may be extracted) is outlined. This component assists with the initial system creation process which can be an especially difficult activity for novices.

1. INTRODUCTION

Fuzzy logic has often been recognized, and even more often *evangelized*, as a useful addition to the modeling tool-boxes of researchers and practitioners in many fields. The extent of this recognition appears to be still growing, thereby increasing the range of potential users and adding still more applied researchers and practitioners without a deep or detailed understanding of fuzzy sets and fuzzy logic to this set.

The question that this paper is concerned with is how software should support these novice users when they are starting to use fuzzy logic models. Obviously their requirements are quite different from those of a fuzzy researcher investigating a practical problem who will generally require additional support more on the application side, if at all.

1.1 Choice of techniques

At the same time that users are moving to fuzzy methods, the developer of a predictive or classification model is faced with an ever widening choice of techniques (and technique options) that are available in standard or commonly use packages; these packages are often intended for applied researchers and practitioners. As well as the traditional statistical models (such as regression, discriminant analysis, and Analysis of Variance), there are more sophisticated techniques for dealing with particular classes of problems including nominal and ordinal logistic regression, resampling techniques, Bayesian networks, and various neural network architectures and training algorithms.

All of these techniques are becoming increasingly accessible to non-specialists through the emergence of

such software packages. This means that difficult to access techniques may be neglected in favor of other more accessible techniques, while still permitting the analyst to explore and contrast a variety of techniques if that is their goal. In short, an increasingly large number of users want, or even demand, that the applications they use are user-friendly, offer a wide variety of options, and automate as much of the model development process as possible (as in wizards, automated file conversion, and network awareness, among others).

User-friendly software is rapidly becoming the norm, so we will not attempt to justify why fuzzy logic software should try to follow this pattern. The other characteristics are however worth a little justification. The breadth of packages in terms of techniques and options can be regarded as driven by curiosity or the desire for publishable/impressive results, or can be seen as sensible scientific practice. The desire for automation is easily observed in the widespread practice of stepwise regression which is seldom appropriate on technical grounds. Therefore any tool aiming for penetrating this market must satisfy these interface and functionality requirements, at the very least.

1.2 Problem characteristics

It also has to be recognized that not all model building exercises are undertaken with the luxury of large quantities of uncontaminated data; this is often necessary for tuning models implemented by regression or neural networks, especially where the data is of high dimensionality. In some cases data quickly becomes outdated, such as marketing data and software development project metrics; in other cases the data is too expensive to collect in large quantities, such as certain chemical and biological processes; and in other situations the data is simply not available, such as studies of rare genetic phenomena.

Also, the advantages of an expert-knowledge driven and relatively transparent modeling process are attractive in their own ways to many practitioners and researchers; sometimes understandability is more important than numerical accuracy. The growth in the fields of knowledge management, organizational memories, and data mining are indicative of the (economic) value placed on developing and preserving knowledge.

2. FUZZY LOGIC MODELS

These data modeling demands favor the use of techniques such as fuzzy logic and case-based reasoning where qualitative factors receive, or at least can receive, more attention, as opposed to strictly data driven techniques. Of course, case based reasoning can be treated as a data driven method, but its intelligent use generally involves expert knowledge to a large degree.

In some respects fuzzy logic also responds well to the first issue raised above of making the technique accessible to the user. Fuzzy logic is almost automatically more accessible, albeit less direct perhaps in its process, than many commonly used techniques ranging from regression to feed-forward neural networks. The use of concepts and associations (via membership functions and rules) is often easier to grasp than the use of equations and function optimization.

As a result of this shift in attitude and practice away from modeling to modeling *and understanding*, many non-specialist courses now include a section on fuzzy set theory and/or logic to better prepare their students for understanding and using these approaches in practice. Similarly, in many fields of application practitioners have begun to investigate the use of fuzzy theory for their particular domain of interest. Fuzzy logic models may complement or even replace other methods, depending on the particular set of circumstances under which the analysis is performed.

Both these classes of *non-specialist*s (students and practitioners) are easily overwhelmed by the many options and methods available to them, especially in some of the more advanced software packages. Similarly, they often fail to understand relatively elementary concepts when presented with algorithms and equations as explanation, for example, the many forms of defuzzification. In the case of students this can be discouraging and lead to the student not properly understanding the material. In the case of practitioners, their demands are much more stringent, both in terms of speed of learning and operation, as well as quality of product.

The two main difficulties implied by the above can be regarded as requiring the software to follow a sensible process model for system generation and explaining the model development and inference processes. Either of these problems, if inadequately resolved, can lead to non-optimal use of the techniques at hand (and thus either set poor examples for others to follow or produce discouraging results). Or the technique may simply be abandoned in favor of other more accessible options.

3. FUZZY LOGIC FOR PROJECT MANAGEMENT

Project management is an area that is well suited to fuzzy logic: data is a rare commodity and is often contaminated, expert judgment is the predominant techniques, and there is a recognition of the need for more structured estimation techniques. It is therefore perhaps surprising that there has not been more work in applying fuzzy logic to project management of software development.

When faced with the task of encouraging software development project managers to make use, when appropriate, of fuzzy logic models [1, 2] it was necessary to recognize that the unavailability of appropriately simplified and graphical tools could present an insurmountable hurdle for such potential users. Similar problems could also be observed in teaching introductory level students the fundamentals of fuzzy logic.

This motivated a three stage approach to educating novices. Firstly, to provide suitable software (which we could not find elsewhere); secondly to develop scenario-based process models for guiding users, and potential users, through the necessary steps without being unduly restrictive; thirdly to provide appropriate (and multimedia) documentation that spanned all levels of expertise.

Our goals for the software were, and still are with respect to new modules, to provide a single integrated environment that would handle data capture and manipulation, model creation, prediction, and the analysis of the results. All of this needed to be accomplished with an intuitive interface, mostly graphically based in order to best serve the user.

4. FUZZYMANAGER

Figures 1 and 2 show screen shots of the FUZZYMANAGER system. These demonstrate the basic functionality and approach taken in the application design. The modeling process has been made as graphical as possible without becoming gratuitous in pursuit of this goal. As can be seen, the FUZZYMANAGER software [3] consists of two main application modules, although several others are planned to be added in the next two years.

The core application is FULSOME (Fuzzy Logic for Software Metrics, again betraying the origins of this project). This provides a simple to use interface to a reasonably complete fuzzy logic environment. In recognition of the learning process that users go though, a large number of warnings (such as 'out of range data', 'no rules fired by a row of data', and 'missing data') are implemented but can be individually turned off.

As well as basic data handling facilities, the process of membership function and rule entry has been made as painless as possible. Membership functions of various types (Triangular, Trapezoidal, Gaussian, Bell-Shaped, and Sigmoidal) can be created automatically (evenly spaced) or tuned individually. As the parameters are updated, the graphical representation is refreshed. Shoulders are provided by setting maximum and minimum values for a variable. All out of range values are scaled to the appropriate limit, simulating shoulders for the upper and lower most functions.

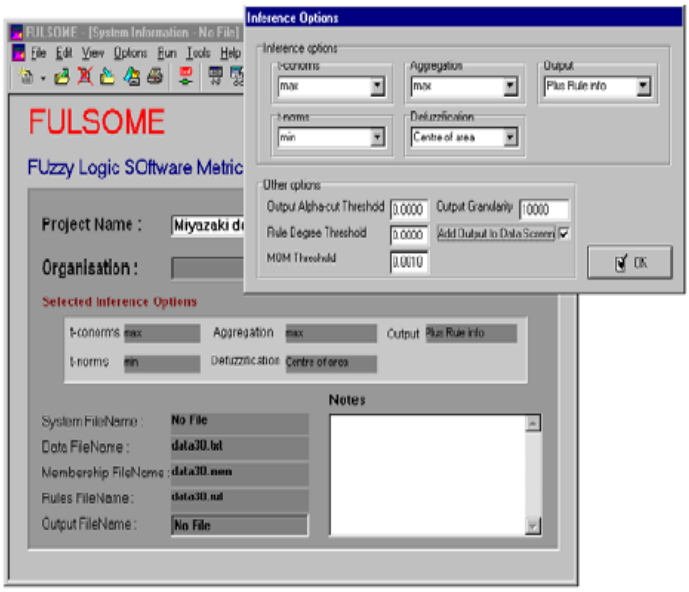
Rule entry is facilitated by a 'combo-box' approach where the variables and labels are selected from lists. This prevents the problems with text file-based fuzzy logic systems where typographical errors can cause many problems for novices. Standard logical operators, ARE and OR, are provided as well as bracketing and rule weights. We have intentionally avoided clause weights since they are seldom useful in our experiences and add another, unnecessary, level of complexity for the user.

Various options for inference, such as center of gravity (COG), mean of maxima (MOM), and center of area (COA) defuzzification; a variety of t-norms and t-conorms for logical operators and aggregation are provided to encourage experimentation. Useful options for applying α-cuts to the output and rule-degree thresholds are provided, which are especially useful when using Gaussian membership functions. Since the inference is entirely numerical, a granularity factor is provided to indicate, at a global label, the number of steps in the membership functions. A larger number will give more accurate results at the cost of speed. This was thought to be an important consideration to allow users with faster or slower hardware than usual to adjust the program to better serve their needs.

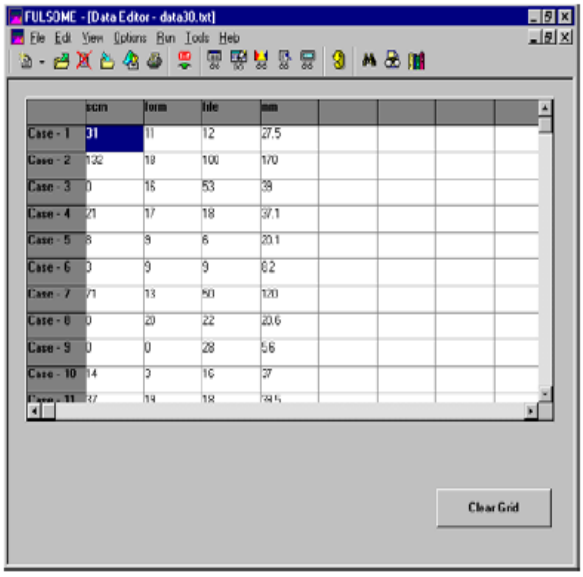
Outputs can be tracked in terms of individual observations (rows in the data grid), rules that were activated by the input variables (and the weighted and unweighted degrees to which they were fired), and clauses within the rules (the degree to which they were satisfied by the appropriate variable's value), the contribution of each rule to the output variable, and finally the defuzzification of the output variable membership function if required. These allows the user to visualize the system's inference process, stepping through each stage to see how it leads to the next and working either top-down or bottom-up depending on their needs.

The other application is the similarly named CLUESOME (Cluster Extraction for Software Metrics) which extracts membership functions and/or rules from raw data. These are found by using one-dimensional fuzzy c-means clustering to determine the centers of the membership functions, and by using multi-dimensional clusters to extract weighted rules based on membership function activations. The membership functions are then defined using the appropriate values for *spread* parameters to ensure sensible coverage of the input space. Rules can be weighted by using a number of operators (sum, max, mean, and bounded versions) on the membership function activations by the rule cluster centers.

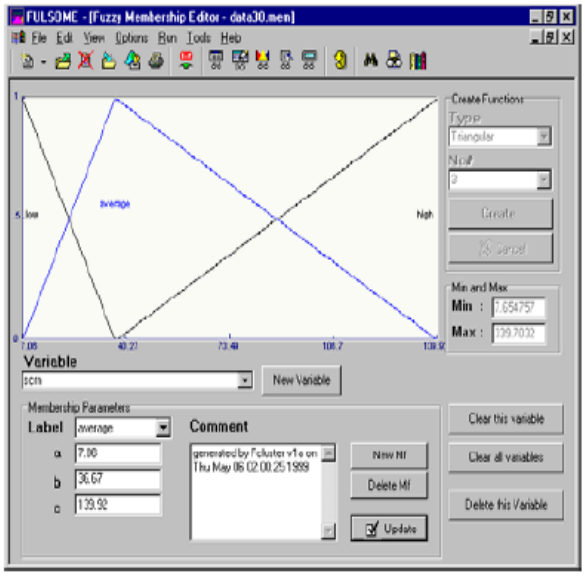
While the resulting rules are certainly not optimal in terms of numerical accuracy for the mapping, they do provide a reasonable start for an expert to tune based on their perceptions. More importantly, the creation algorithm can be easily explained graphically and appears reasonable to *fuzzy novices*. This has been a vital factor in



(a) System screen (which collects files into systems for easier access)

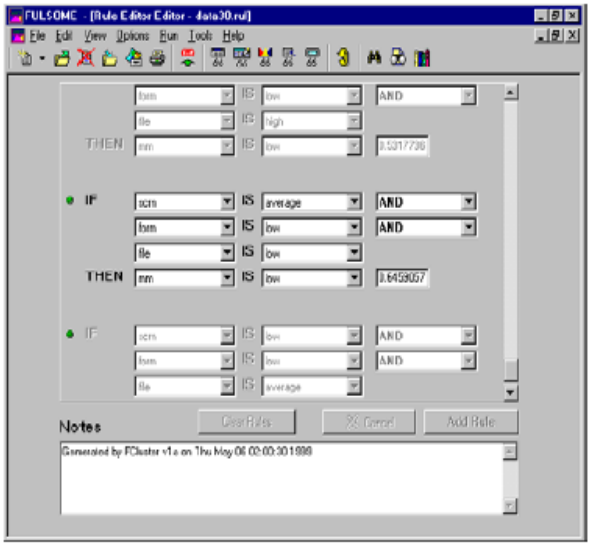


(b) Data editor

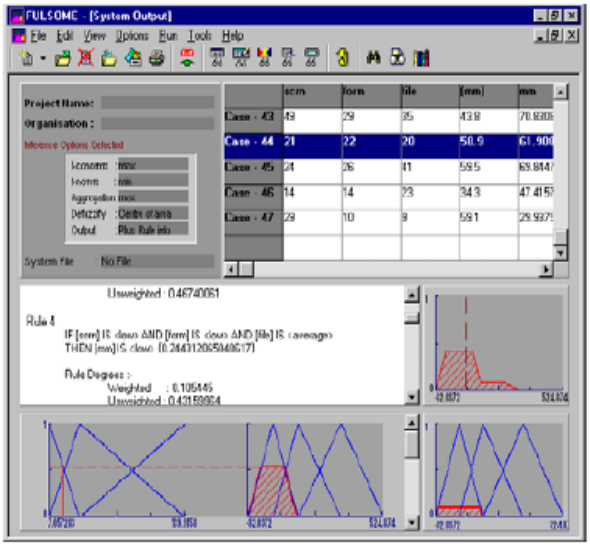


(c) Membership function editor

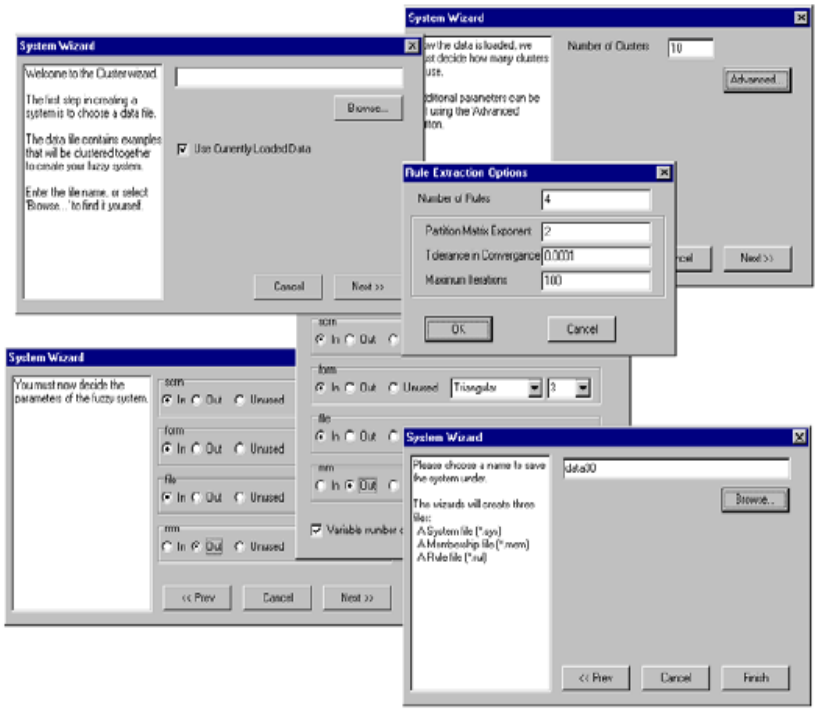
Figure 1: Structure of the FUZZYMANAGER system (part one)



(a) Rule editor



(b) Output screen showing inference traces



(c) CLUESOME Wizard

Figure 2: Structure of the FUZZYMANAGER system (part two)

gaining acceptance from pragmatic project managers who are naturally suspicious of *black box* models.

Initially the software was developed under Microsoft Windows, since this was the most common platform for our initial group of users. The *engine* code has, however, been written in libraries using standard C^{++}. Therefore, porting the libraries to other platforms is relatively trivial.

5. CONCLUSIONS

In this paper we have very briefly described an application suite called FUZZYMANAGER that has been developed with the goal of providing an accessible, and yet still capable, system for fuzzy logic inference. The system is currently being evaluated for use in introductory courses in several universities around the world, and is being used in trials by various New Zealand software developers. The feedback from these two groups will be used to further refine the existing applications.

DOWNLOADING

The software can be downloaded from http:// smrl.otago.ac.nz/downloads/software/ fuzzymanager/. It is completely free for both commercial and non-commercial use, although we do appreciate feedback from our users. Source code may be made available on request to fuzzymanager@ infoscience.otago.ac.nz.

ACKNOWLEDGMENTS

The authors would like to thank Richard Kilgour for his part in developing the CLUESOME application and several suggestions about the FULSOME module. We would especially like to thank Nikola Kasabov who's FuzzyCOPE, CBIS, and CBIIS applications were an inspiration to the software developed.

REFERENCES

[1] A. Gray and S. MacDonell. Applications of fuzzy logic to software metric models for development effort estimation. In *Proceedings of the 1997 Annual meeting of the North American Fuzzy Information Processing Society -NAFIPS'97*, pages 394–399. IEEE, 1997.

[2] A. Gray and S. MacDonell. Fuzzy logic for software metric models throughout the development life-cycle. In *Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society -NAFIPS*, pages 258–262. IEEE, 1999.

[3] S. MacDonell, A. Gray, and J. Calvert. Fulsome: A fuzzy logic modelling tool for software metricians. In *Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society -NAFIPS*, pages 263–267. IEEE, 1999.