10.2  
Some Design Methods  
551

• By combining the SVD–QR method with a derivative-based or particle-based  
method, one can design all of the parameters of an IT2 fuzzy system including  
the number of the most signiﬁcant rules, M0. The following iterative design  
method can be very successful:

1. Fix the number of rules, M, at a reasonable value.  
2. Use a derivative-based or particle-based method to design all the antecedent  
and consequent FOU parameters.  
3. Apply the SVD–QR method to the results of the derivative-based or  
particle-based method to determine M0\M IT2 FBFs.  
4. Renormalize the IT2 FBFs and re-compute the linear combining parameters  
using least-squares.  
5. If performance is acceptable, STOP. Otherwise, return to Step 2 for a  
re-tuning of the antecedent and consequent parameters.

• By applying the SVD-QR method to the IT2 FBF matrix that can be created  
after IT2 Zadeh rules are obtained from the IT2 WM method.  
• By using an evolutionary or bio-inspired optimization method that is set up not  
only to optimize FOU parameters, but also other things such as [e.g., Rutkowski  
(2004)]: which antecedents to use as well as their number (i.e., p), the number of  
linguistic terms for each variable (i.e., Q1; . . .; Qp), the number of rules (i.e., M),  
the t-norm used (i.e., minimum or product), Mamdani product or minimum, and  
the type-reduction method (e.g., height or COS).

10.2.7  
Remarks

10.2.7.1  
General Remarks

The following objection to optimal IT2 fuzzy system designs is sometimes raised:  
Because an IT2 fuzzy system is described by more parameters than is a T1 fuzzy  
system, it is unfair to compare the performance from such an IT2 fuzzy system with  
the T1 fuzzy system, that is, it is only fair to compare optimal designs for IT2 and  
T1 fuzzy systems that have exactly the same number of parameters. Interestingly, a  
similar objection is not raised when optimal designs are compared for a T1 fuzzy  
system and a non-fuzzy system, in which the T1 fuzzy system has more design  
degrees of freedom than the non-fuzzy system. The design approach advocated in  
this book is one that ﬁrst begins with a T1 fuzzy system and tries to achieve the  
desired performance. It is only when such desired performance cannot be met that  
this book advocates moving up to an IT2 fuzzy system.  
It is worth restating some of the general remarks that are given in Sect. 4.2.7 but  
in the context of IT2 fuzzy system designs.  
When an IT2 fuzzy system is going to be used as part of a consumer (or military)  
product then it should be designed to meet pre-speciﬁed performance speciﬁcations

10.3  
Case Study: Forecasting of Time-Series  
563

10.3.2.2  
One-Epoch Combined Derivative-Based and SVD-QR Design

In this fuzzy system design, the derivative-based (steepest descent) and SVD–QR  
methods were combined. To do this the steepest descent method was used for just  
one epoch of training after which the SVD–QR method was applied to its results.  
As in the previous section, ﬁve Mamdani fuzzy system forecasters were designed:  
singleton T1, non-singleton T1, singleton IT2, T1 non-singleton IT2, and IT2  
non-singleton IT2. All of the previous section’s discussions about number of data  
points, training points, testing points, number of rule antecedents, number of fuzzy  
sets for each antecedent, number of rules, choices for antecedent, consequent and  
input measurement MFs, initial choices for MF parameters (using the totally  
independent design approach), and evaluation by means of RMSE formulas remain  
the same for the present designs.  
50 Monte Carlo realizations were run for each of the ﬁve designs, and for each  
realization the fuzzy system was tuned before rule-reduction using a simple steepest  
descent algorithm, but only for one epoch. Each fuzzy system was then rule-reduced  
using the appropriate SVD–QR method (see discussions about SVD–QR designs in  
Sects. 4.2.4 and 10.2.4). The number of rules to be retained was established by using  
a threshold, c (set arbitrarily to 1), for the singular values that were computed for the  
SVD of a FBF matrix [e.g., (10.29) and (10.30), making use of the discussions on  
how to use these FBF matrices for an IT2 Mamdani fuzzy system with COS  
type-reduction + defuzziﬁcation, that is, given at the end of Example 10.9]. Let sj  
denote those singular values; then ^  
r was chosen such that s^  
r 1.  
RMSEs were computed both before and after rule-reduction. Results are sum-  
marized in Tables 10.8 and 10.9. Observe, from Table 10.9 that there is a very  
substantial reduction in the number of rules, from 16 to anywhere from 4 to 9.  
Unfortunately, there is an accompanying degradation in RMSE performance, as can  
be seen from the entries in Table 10.8. Our next design attempts to both improve  
the rule-reduced RMSEs and to further reduce the number of rules.

10.3.2.3  
Six-Epoch Iterative Combined Derivative-Based  
and SVD-QR Design

Next, the designs of ﬁve fuzzy system forecasters are compared for the Mackey–  
Glass time-series using an iterative version of combined derivative-based and  
SVD–QR methods. As in the previous sections, ﬁve Mamdani fuzzy system  
forecasters were designed: singleton T1, non-singleton T1, singleton IT2, T1  
non-singleton IT2, and IT2 non-singleton IT2. Additionally, all of Sect. 10.3.2.1’s  
discussions about number of data points, training points, testing points, number of  
rule antecedents, number of fuzzy sets for each antecedent, number of rules, choices  
for antecedent, consequent and input measurement MFs, initial choices for MF  
parameters (using the totally independent design approach), and evaluation by  
means of RMSE formulas remain the same for the present designs.

10.4  
Case Study: Knowledge Mining Using Surveys  
581

does the EIA. An interesting feature of the HMA is that the word FOUs are  
completely normal (i.e., both their UMF and LMF are normal T1 FSs), whereas  
only the UMFs from the IA and EIA are normal T1 FSs.

Regarding the Engine of the Perceptual Computer, Mendel and Wu (2010,

Chaps. 5 and 6) describe two kinds of engines—if-then rules and novel weighted  
averages. For if-then rules, they advocate determining a ﬁring level rather than a  
ﬁring interval, by using the Jaccard similarity measure for IT2 FSs (Exercise 7.46),  
so that the ﬁnal combined IT2 FS is more similar looking to an application’s  
codebook FOU than is the ﬁnal combined IT2 FS obtained when ﬁring intervals are  
used.

Novel weighted averages range from the IWA (Sect. 8.2) to the fuzzy weighted

average (which only uses T1 FSs—see Exercise 8.15) to the linguistic weighted  
average (which uses IT2 FSs, or a mixture of T1 and IT2 FSs). The latter is a  
weighted average, where weights and evaluations are linguistic terms, whose FOUs  
can be modeled, e.g. by using the HMA. Another very powerful NWA is the  
linguistic weighted power mean (Rickard et al. 2011, 2013).

Regarding the Decoder of the Perceptual Computer (Mendel and Wu 2010,

Chap. 4), similarity and subsethood (Exercise 7.47) play very important roles.

An important aspect of the Perceptual Computer is that the complete vocabulary

of all of the words that are used in an application must be established before IT2 FS  
models are found for the words. The size of the vocabulary for a linguistic variable  
affects the calibration of the fuzzy sets. If, for example, only three linguistic terms  
are used to describe Proﬁtable, namely {hardly proﬁtable, moderately proﬁtable,  
fully proﬁtable}, then their fuzzy sets will look very different from their fuzzy sets  
when the following six terms are used: {barely proﬁtable, hardly proﬁtable,  
somewhat proﬁtable, moderately proﬁtable, fully proﬁtable, extremely proﬁtable}.  
This is because the term barely proﬁtable now appears before hardly proﬁtable, and  
the term extremely proﬁtable now appears after fully proﬁtable. So, knowing the  
complete vocabulary for all of the linguistic variables is crucial to the proper  
modeling of the words in an application.

Another interesting aspect of the Perceptual Computer is that it can only be

interacted with using words that are in the codebook. When words are modeled as  
IT2 FSs, and, e.g., the Engine is if-then rules, then one is always in the situation of  
IT2 non-singleton fuzziﬁcation! That is the bad news. The good news is that since  
the vocabulary and codebook are known ahead of time, all possible ﬁring intervals  
can be pre-computed and then stored in a look-up table.

Finally, Chap. 6 in Mendel and Wu (2010) gives all of the details for a

Perceptual Computer Flirtation Advisor, using the same data that have been used in  
this book.