Type-2 Fuzzy Logic System

Applications for Power Systems

Iván Castro León

Newcastle University

A thesis submitted for the degree of

Doctor of Philosophy

December 2016

To Cristina

Abstract

In the move towards ubiquitous information & communications technology, an opportunity for further optimisation of the power system as a whole has arisen. Nonetheless, the fast growth of intermittent generation concurrently with markets deregulation is driving a need for timely algorithms that can derive value from these new data sources. Type-2 fuzzy logic systems can offer approximate solutions to these computationally hard tasks by expressing non-linear relationships in a more flexible fashion. This thesis explores how type-2 fuzzy logic systems can provide solutions to two of these challenging power system problems; short-term load forecasting and voltage control in distribution networks. On one hand, time-series forecasting is a key input for economic secure power systems as there are many tasks that require a precise determination of the future short-term load (e.g. unit commitment or security assessment among others), but also when dealing with electricity as commodity. As a consequence, short-term load forecasting becomes essential for energy stakeholders and any inaccuracy can be directly translated into their financial performance. All these is reflected in current power systems literature trends where a significant number of papers cover the subject. Extending the existing literature, this work focuses in how these should be implemented from beginning to end to bring to light their predictive performance. Following this research direction, this thesis introduces a novel framework to automatically design type-2 fuzzy logic systems. On the other hand, the low-carbon economy is pushing the grid status even closer to its operational limits. Distribution networks are becoming active systems with power flows and voltages defined not only by load, but also by generation. As consequence, even if it is not yet absolutely clear how power systems will evolve in the long-term, all plausible future scenarios claim for real-time algorithms that can provide near optimal solutions to this challenging mixed-integer non-linear problem. Aligned with research and industry efforts, this thesis introduces a scalable implementation to tackle this task in divide-and-conquer fashion.

Declaration

I hereby declare that this thesis is a record of work undertaken by myself, that it has not been the subject of any previous application for a degree, and that all sources of information have been duly acknowledged.

Acknowledgements

First, I would like to thank Professor Phil Taylor as without his help, this work would not be have been possible.
I would also like to acknowledge my colleagues at Newcastle and Durham; Peter Davison, Jialiang Yi, Pengfei Wang, Calum Cameron, James King, David Greenwood, Pàdraig Lyons for sharing a lot of interesting discussions, and specially to Emmanouil Loukarakis and Varvara Alimisi for their unconditional support.
I would also like to thank my family as I know that regardless where I am, I know they are by my side.
Finally, thanks to Cristina, as this thesis is for you.

List of Publications

Conference papers

- Castro León, I. and Taylor, P.C., 2013, October. Towards autonomic control in decentralised power systems via distributed type-2 fuzzy systems. In IEEE PES ISGT Europe 2013 (pp. 1-5). IEEE.
- Castro León, I. and Taylor, P.C., 2015, June. Memetic Type-2 Fuzzy System Learning for Load Forecasting. In 2015 Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology (IFSA-EUSFLAT-15). Atlantis Press.

Contents

List of Figures

Acronyms

AI Artificial Intelligence. 3

ANM Active Network Management. 1

BFGS Broyden-Fletcher-Goldfarb-Shanno Algorithm. 18

DG Distributed Generator. 1

FLS Fuzzy Logic System. 1

FOU Footprint Of Uncertainty. 9

ICT Information Communication Technology. 2

MAPE Mean Average Percentage Error. 34

MF Membership Function. 9

NN Neural Network. 3

OLTC On-Load Tap Changer. 4

STLF Short-Term Load Forecasting. 1

SVR Support Vector Regressor. 3

Nomenclature

x_i	i-th primary input FLS variable
$\mathbf{x}$	Vector of primary input FLS variables
p	Number of primary input FLS variables
u_i	i-th secondary input FLS variable
X_i	Universe of discourse of the i-th primary input FLS variable
J_{x_i}	Primary membership of the i-th primary input FLS variable
$\mu_{F_{l,i}}$	Membership function of the type-1 fuzzy set for the l-th rule and i-th input
$\underline{\mu}_{F_{l,i}}$	Lower membership function of the type-1 fuzzy set for the l-th rule and i-th input
$\overline{\mu}_{\tilde{F}_i}$	Upper membership function of the type-1 fuzzy set for the l-th rule and i-th input
$\star$	Fuzzy set t-norm operator
$\vee$	Fuzzy set t-conorm operator
$m_{l,i}$	Gaussian membership function mean of the l-th rule and i-th input
$\sigma_{l,i}$	Gaussian membership function deviation of the l-th rule and i-th input
$\underline{\sigma}_{l,i}$	Gaussian membership function lower deviation of the l-th rule and i-th input
$\overline{\sigma}_{l,i}$	Gaussian membership function upper deviation of the l-th rule and i-th input
$h_l(\mathbf{x})$	Fuzzy logic system consequent of the l-th rule
$c_{l,l}$	Left bound of the singleton FLS consequent of the l-th rule
$c_{l,r}$	Right bound of the singleton FLS consequent of the l-th rule
$c_{l,i}(x_i)$	Multivariate linear regression coefficient of the fuzzy logic system consequent of the l-th rule and and i-th input

R_l Rule l-th

M Number of rules

$\underline{f}_l$ Lower firing level of the l-th rule

$\overline{f}_l$ Upper firing level of the l-th rule

L Left switching point

R Right switching point

c_l Left bound of type-reduced set

c_r Right bound of type-reduced set

y Output FLS variable

$I_{X_i,S,Y}$ Information-interaction between the feature X_i, a set of features S, and the output Y

$H(X_i)$ Entropy of X_i

$H(X_i;Y)$ Mutual information between the feature X_i and Y

$H(X_i;X_j \mid Y)$ Conditional mutual information between the features X_i and X_j given Y

$\psi(x)$ Digamma function

β Redundancy parameter

γ Condicional redundancy parameter

ϵ_k Variable that represents the maximum norm in all dimensions of the k-th nearest neighbour

n_{x_i} Number of points within ϵ_k in the X_i space

d_{X_i} Absolute deviation of X_i for a given set

Z_{X_i} Z-score of X_i for a given set

pr_{cs} Probability of ruleset crossover

pr_{cr} Probability of rule crossover

pr_m Probability of mutation

A_j FLS A_j

$\mathcal{P}(A_j, D)$ Fitness of the FLS A_j given the dataset D

D Dataset

N Number of samples

α Complexity penalty

$S(A_1, A_2)$ Similarity between A_1 and A_2

B Hessian approximation

d Search direction

λ Step size

τ_{max} Convergence tolerance

i_{max} Number of iterations

$w_{A_j,l,i}$ Antecedent/Consequent parameter of the i-th input of the l-th rule of the FLS A_j

v_i Voltage deviation in the i-th node

1 Introduction

Evolution of power systems towards actively managed systems has led to an increasing need for two particular power system applications; short-term load forecasting (STLF) and active network management (ANM) at distribution level. These two distinctive targets have different associated issues that can be tackled via type-2 fuzzy logic systems (FLSs). Firstly, the non-stationary behaviour of load time-series in mean and variance with multiple seasonalities in conjunction with many influencing factors, such as weather, economic activity or social habits. Secondly, the need for timely solutions to mixed-integer non-linear problems. This chapter provides an overview and the main research objectives of this thesis.

1.1 Background

Power systems are traditionally operated in a passive fit-and-forget fashion with a top-down design where power flows are unidirectional from the transmission level to end-users. However, this is changing. First, the push towards emissions reduction has led to a significant growth in intermittent distributed generation (DG), which can lead bi-directional power flows and fluctuation in voltage, pushing the grid closer to its statutory limits.

This drives a need for active network management (ANM) [1] to mitigate the effects of these non-dispatchable generators. Second, not only generation is changing, electrification of heat and transport will steadily increase consumption and its profile will change with higher peaks [2]. Higher peaks directly translate into capacity problems. An option to avoid these, is to forecast them to make the necessary arrangements to mitigate their effects. Thus, it is important to now not only their scale but also when they will occur. Third, energy markets are also changing, and potentially exposing end-users to price signals can lead to additional flexibility [3]. This in turn increases the need for accurate short-term forecasts and efficient system-wide coordination. Last but not least, the uptake of information and communications

technology (ICT) [4, 5, 6, 7] enables this change, as new data sources are available. All these, forces the power system to move towards smarter ways of meeting the end-users requirements. To improve network utilisation, reduce operating costs, increase renewable energy penetration or allow an uptake in new types of demand, contributions towards STLF and ANM with a particular focus on voltage control at the distribution level are required. This thesis focuses on the development of these two particular aspects, and does it by means of type-2 FLSs. In the following we elaborate on the reasons why we opt for these types of systems.

1.2 Short-Term Load Forecasting

Load forecasting heavily influences electrical power systems planning and operation in different time-frames: Long-term, medium-term and short-term [8]. Long-term forecasts define planning strategies according to future demand and policy (i.e. one/several-year/s-ahead). Medium-term support operation and maintenance (i.e. up to one/several-month/s-ahead). Short-term forecasts (i.e. one-day-ahead 30-minutes-resolution) [9], are a key input for economic and secure power systems [10, 11]. There are many tasks that require a precise determination of the future short-term load with different time-scales and on different hierarchies [12].
Not only, transmission system operators require precise forecasts to make unit commitment decisions, reduce spinning reserves or schedule maintenance plans accordingly. Furthermore, they may use load forecasts to prepare backup plans under-predictions lead to bring new units on-line or shed loads due to insufficient reserve capacity.
But also, the intensification of electricity market deregulation [13, 14, 15, 16] causes a need for accurate STLF. In wholesale markets, energy prices may be affected by fluctuations in demand and/or weather conditions, and notably boosted during on-peak periods. Among other market players, suppliers require one-day-ahead as the basis for their trading strategies to ensure they have purchased accurately for their customers, otherwise they may end up making a loss in the intra-day market for or even being penalised via cash-out penalties after the gate closure. Aggregators may need to schedule their flexibility according to load forecasts to provide balancing services to the former. Therefore, accurate STLF becomes essential for generators, utilities and consumers and any forecasting inaccuracy can be directly translated into

their financial benefits [17, 18, 19]. The importance of STLF will grow as systems get optimisation becomes wider, as it is a key enabler.
However, obtaining reliable models for STLF is still a challenging task (i.e. non-stationary time series in mean and variance with multiple seasonalities in conjunction with many influencing factors, such as weather, economic activity or social habits).
From the methods that have been proposed in the literature for STLF [20], two broad categories can be distinguished: traditional methods [21, 22] (e.g. Auto-regressive integrated moving average, Kalman filtering models, Box-Jenkins models, exponential smoothing or regression models, among others); and artificial intelligence (AI) methods [23, 24, 25, 26, 27, 28], such as neural networks (NNs), support vector regressors (SVRs) and FLSs. The former are restricted by their linearity, although they are significantly more efficient in terms of computation costs. The latter are universal approximators and can express non-linear relationships in a more generalised manner. Even though, there are no consistent comparative results which indicate their supremacy over traditional methods [29], AI-based approaches are extensively used for short-term load prediction [30, 31].
In [25], an electricity-domain AI-focused comparative study was performed, indicating that interval type-2 FLSs outperform type-1 FLSs and NNs in terms of generalisation power and forecasting error. This improved performance arises from type-2 FLSs enhanced capability to handle imprecise information [32]. This is achieved through the use of interval type-2 fuzzy sets rather than "*crisp*" type-1 fuzzy sets. Furthermore, in interval type-2 FLSs, a smaller rule-base and fewer fuzzy sets can be used to define the entire input/output range (i.e. universe of discourse) due to a better trade-off between modelling accuracy and complexity [32].
Nonetheless, type-2 FLSs in [25] are still constrained as there is room for improvement in terms how these are defined from selecting their inputs, to establish their inner structure and parameters.

1.3 Voltage Control in Distribution Networks

Fast growth of intermittent generation in low- and medium-voltage distribution systems concurrently with arising development of active demand and new types of demand is radically transforming the role distribution system operators are playing [33, 6]. The

move towards a low-carbon economy is pushing the grid status even closer to its operational limits. Distribution networks are becoming active systems with power flows and voltages defined not only by load, but also by generation. As a consequence, problems like voltage excursions and thermal bottlenecks will become more frequent. These two, voltage excursion and overloading, are inter-related. However, they operate in different time-scales and solving one does not necessarily solve both. Thus, while it is not yet absolutely clear how power systems will evolve in the long-term, all plausible future scenarios indicate a need of improved efficiency and harnessing available flexibility [34]. Moreover, given that the main objective of every operator is to maximise the use of their network assets, that is to say, to guarantee that its previous and present investments are used in the best possible and reachable fashion. Modernising control via smart grid technologies [35, 36] can provide an alternative of meeting customer requirements and optimise network operation.
Traditionally, voltage was controlled in distribution networks through the network design so voltage problems do not arise. There is minimal control and always located near high-voltage levels. However, as it has been outlined earlier, distribution system operators will have difficulties in maintaining the voltage within the statutory limits as the generation/demand ratio changes and becomes erratic.
While network reinforcements remain important to address these and increase capacity, they may be unaffordable when facing the low-carbon transition since also many assets are getting close to the end of their predicted life. Furthermore, operators need to justify the cost in terms of return of investment [37].
Nonetheless, several are the active voltage control devices that already may be present in distribution networks [1, 38] (e.g. from on-load tap changers to regulators in synchronous generators). Commonly, the problem is approached with no direct coordination, where each voltage regulation device by means of local measurements and a voltage target defines its control actions. For instance, on-load tap changers (OLTCs) operate in conjunction with automatic voltage control relays. The relay continuously monitors the secondary voltage and current to adjust the transformer's tap ratio and keep the measured voltage at a given target. Two modes are used in these, voltage fixed set-point and load drop compensation. The latter pursues to balance the voltage target under different load scenarios to ensure the operation is appropriate, whilst the former acts regardless the loading level.
An alternative, as networks were mainly designed for load-only ones, especially at

low-voltage level, is to limit the real power output of the distributed generators, as distribution networks are mainly resistive, particularly in low-voltage level, DGs can significantly rise voltage. However, this curtailment may impact 'first-on-last-off' contractual policies [38] between the generator and the operator and limits the significant contribution to CO_2 emissions minimisation [39] of distributed generators. Although, controlling their reactive power output can provide voltage support, but given the R/X ratio of distribution networks, this will be insufficient in many cases [40]. Therefore, all these constraints or penalties are limiting the value of DGs.
Nevertheless, due to the complexity of the existing voltage control problem, finding localised solutions may also cause an increase in power losses or dynamic interactions inside the network, as OLTCs and DGs can affect each other's operation. Therefore, voltage control without direct communication is also tackled by means of time delays [41]. Hence, devices operate sequentially to avoid interactions. The order of operations between devices is predefined. Downstream devices in radial networks as have increasing delays. Hence, devices closer to the end of the feeder operate last [41].
To further optimise the previous approaches, by means of a few local measurements, voltage targets can be actively determined. SuperTAPP n+ [42] uses an additional current measurement on the feeder with DGs to calculate the load share ratio between feeders with and without generation to estimate a new target. Although it represents an improvement, it has several drawbacks (e.g. assumes load and generation patterns are fixed, relies on at least one load-only feeder being available and assumes network topologies are static). GenAVC [42] relies in network state estimation through remote measurement units. However its accuracy of the estimates is correlated with the number of measurement points. It also assumes fixed topology and load predictability.
As result, all these uncoordinated or localised control strategies are often insufficient in solving complex voltage control problems and system wide solutions are necessary [43, 44] to overcome the above mentioned issues. These system wide approaches due to the ubiquity of ICT can now be more easily deployed [5] allowing for further optimisation, and therefore this thesis will explore this technical opportunity.
Hence, given a network model [45] and a set of conditions, it is required to find a solution that keeps all nodes within the statutory limits whilst limiting the number of operations in the OLTCs and the curtailment of real power in DGs, all in timely

fashion. Thus, the voltage control problem via direct coordination, represents a challenging mixed-integer non-linear problem. Control devices can be either discrete (e.g. OLTCs, capacitor banks or switchable loads) or continuous (e.g. storage or DGs among others). This, given the size of distribution networks, can lead to intractability in terms of time with conventional optimisation approaches. Heuristic approaches are common in the area and appear to be well suited to this voltage control problem. In [46, 47] a sensitivity-based greedy search algorithms are proposed which moves towards the most promising candidate solution until convergence. However, this local search approach heavily depends on the starting point and may get stuck in a local minima. Blind search algorithms, like evolutionary ones [48, 49, 50, 51], case-based reasoning systems [43] or NNs [52] have been also used for this purpose, although performing a global search can avoid local minima issues, the lack of network knowledge makes generally these algorithms as impractical as the conventional optimisation approaches due to the time required to find a solution. Other soft-computing approaches, particularly using type-1 FLSs and/or multi-agent systems are [53, 54, 55, 56, 57, 58]. Type-1 FLSs are a proven methodology for dealing with imprecision and their the key advantage is their ability to solve non-linear problems and analyse uncertain and qualitative data associated with the process. All these approaches, appear to be well suited to this application. However, as different sources of uncertainty that may affect the FLS definition [32] will be increased, any type-1 approach is limited. Thus, it is necessary to move forward and use fuzzy sets characterised by fuzzy membership functions like in [59], ergo, use type-2 FLSs which were introduced in 1975 by Zadeh [60].

1.4 Type-2 Fuzzy Logic Systems

FLSs are considered a standard when dealing with imprecision with many power system applications (e.g. power electronics, frequency control or energy management systems among others) [61, 62, 63]. Type-2 FLSs are an emerging paradigm that seeks to overcome the traditional type-1 FLS limitations through a more generalised form [32, 59].

This thesis will exploit this ability to handle imprecision and non-linear relationships to tackle the aforementioned problems. In the power systems scenario, when the target is to handle the customer participation in the system, changing goals or effects of

stochastic generation, patterns may not be certain at all (e.g. environmental conditions may change or applying the same action may not always lead to the same result). For instance, even designing an algorithm to control the current scenario with low penetration of DGs; it is not easy to define the exact functions and parameters due to the environment variability and the associated uncertainties. It could be the case that the chosen and totally certain functions are not acceptable anymore. Also, when trying to develop a forecasting model, complex relationships may not be captured by absolutely defined systems. All these limitations in the ability to model will cause degradation in their predictive performance. Also, an algorithm within this context has as well to be robust in the face of corrupted or noisy measurements or to the unreliability of the data used to tune the parameters, ergo, the system has to be fault-tolerant.

Hence, these two problems bring different challenges. In the case of the load forecasting, when one seeks accuracy improvements, type-2 FLSs are constrained by the way they are defined. Despite the fact that they are universal approximators, if the selected inputs, number of rules or parameters are not appropriate, a loss in predictive performance will arise. On the contrary, in the case of voltage control, flexibility and scalability are the main targets, even if the result action is not optimal. Obtaining solutions that solve the voltage excursion in a timely although near-optimal manner is essential.

1.5 Research Objectives

This thesis, by means of type-2 FLSs and their ability to handle imprecision and non-linear relationships, contributes in two distinctive power systems applications, STLF and voltage control in distribution networks.

1. STLF is a challenging problem due to its non-stationary nature and many influencing factors. Thus, the first objective is to examine and establish how type-2 FLSs should be designed, from beginning to end for STLF applications (i.e. from selecting the input variables to establishing their number of rules and defining how the parameters should be optimised in an automated fashion).

2. Voltage control in distribution networks is a mixed-integer non-linear problem where heuristics are commonly used to obtain timely solutions. Type-2 FLSs

have depicted performance in similar scenarios. Therefore the second objective is to develop suitable approaches to cope with this task.

1.6 Thesis Outline

- **Chapter 2** contains an introduction to type-2 FLSs. Provides insights on their inner structure, design considerations and summarises their key differences and advantages when compared with traditional type-1 FLSs.
- **Chapter 3** focuses on how type-2 FLSs should be designed, from beginning to end, for STLF applications. That is to say, from selecting the input variables to defining how the parameters should be optimised. As result, two novel techniques are introduced, an information interaction feature selector and a memetic algorithm. Both algorithms are tested on market-level samples and compared against common combinations of feature selectors, fuzzy logic learning schemes and other AI approaches.
- **Chapter 4** investigates voltage control in distribution networks. Firstly, it provides an overview on literature trends, to then focus on how type-2 FLSs can contribute to coordinated voltage control in distribution networks. In pursuit of near-optimal real-time solutions, a hierarchical implementation of type-2 FLSs is introduced, tested and evaluated against other common heuristics.
- **Chapter 5** concludes this thesis, summarises the main contributions, broader implications and key findings. Finally, it identifies research questions that could effectively extend this work.

2 Type-2 Fuzzy Logic Systems

FLSs are considered a standard when dealing with imprecision. As universal approximators, they have many engineering applications. This chapter provides a background on fuzzy logic, and in particular on the generalised type-2 FLSs. Including their inner structure and design considerations. Also, illustrates their advantages when compared to traditional type-1 counterparts.

2.1 Introduction

Introduced by Zadeh in [64], fuzzy logic is a multi-valued type of logic that contrary to boolean logic, establishes a continuous range of truth values between 0 and 1. This ability to handle partial truth, has made fuzzy logic traditionally seen by the AI community [65] as an approach for managing uncertainty and has numerous applications in power systems [61, 62, 63].

Fuzzy sets are defined by their membership functions (MFs), which depict a distribution of possibilities [66]. Thus, MFs, are a key element in their definition. Traditionally, FLSs are type-1 FLSs, that is to say, their sets are absolutely determined in shape and position (Figure 2.1). Nonetheless, as one can easily point out, being certain in their definition confronts their connotation of imprecision.(i.e. all their degrees of truth are crisp values). Any uncertainty will be translated to certain MFs which will be unable to directly handle them [32, 67] causing limitations to modelling and minimising their effects. Therefore, the use of type-1 fuzzy sets in changing or challenging environments produces an inadequate performance, or the a need to frequently tune the control parameters [68, 69, 70]. As Figure 2.2 shows, type-2 fuzzy sets are characterised by their footprint of uncertainty (FOU) [32, 67, 71] which blurs the exact position and shape of a type-1 MF. Given the aforementioned limitations, type-2 fuzzy sets, characterised by non-crisp MFs [32, 67] are increasingly gaining attention in the literature [72].

However, due to its complexity and computational cost, here like in all real

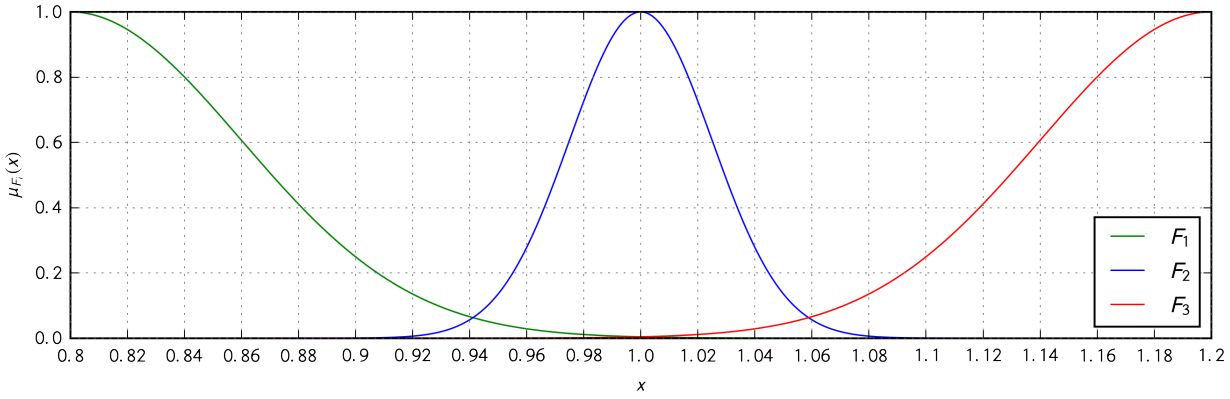

Figure 2.1: Type-1 Fuzzy Sets

applications of type-2 fuzzy logic [72, 73] a special case of type-2 is used to avoid implementation obstacles, which is denoted as interval type-2 [32]. Therefore, two MFs per set will bound this FOU. Thus, during the thesis type-2 is used to refer to these interval type-2 FLSs.

The rest of the chapter is organised as follows: Section 2.2 and 2.3 provide an overview of fuzzy sets and FLSs [32, 59, 74, 75] with a particular focus in the aspects used in this work. Finally, in Section 2.4 conclusions are drawn.

2.2 Fuzzy Sets

A type-1 fuzzy set is denoted by F, and represented as a set of ordered pairs [32],

$$F = \{x, \mu_F(x)) \mid x \in X\} \tag{2.1}$$

where $\mu_F(x)$ is a MF which establishes graded degrees of truth between 0 and 1 over the variable x in the universe of discourse X which represents its operational range. To illustrate this, one can use Figure 2.1 as a voltage control example. The variable x represents the per unit voltage for a given node of a network. Its state is defined by three type-1 fuzzy sets F_1, F_2, and F_3 which denote if the voltage is low, acceptable or high through their crisp MFs $\mu_{F_1}(x)$, $\mu_{F_2}(x)$, and $\mu_{F_3}(x)$. These MFs are gaussian ones (i.e. $N(m_F, \sigma_F; x)$) with parameters $m_{F_1} = 0.8$, $m_{F_2} = 1.0$, $m_{F_3} = 1.2$, and $\sigma_{F_1} = 0.06$, $\sigma_{F_2} = 0.025$, $\sigma_{F_3} = 0.06$ which denote the means and deviations respectively. Thus, for $x_n = 0.96$, $\mu_{F_1}(0.96) = 0.03$, $\mu_{F_2}(0.96) = 0.28$ and $\mu_{F_3}(0.96) = 0$. That is to say, the voltage at 0.96 per unit is low with 0.03 and acceptable with 0.28 degrees of truth.

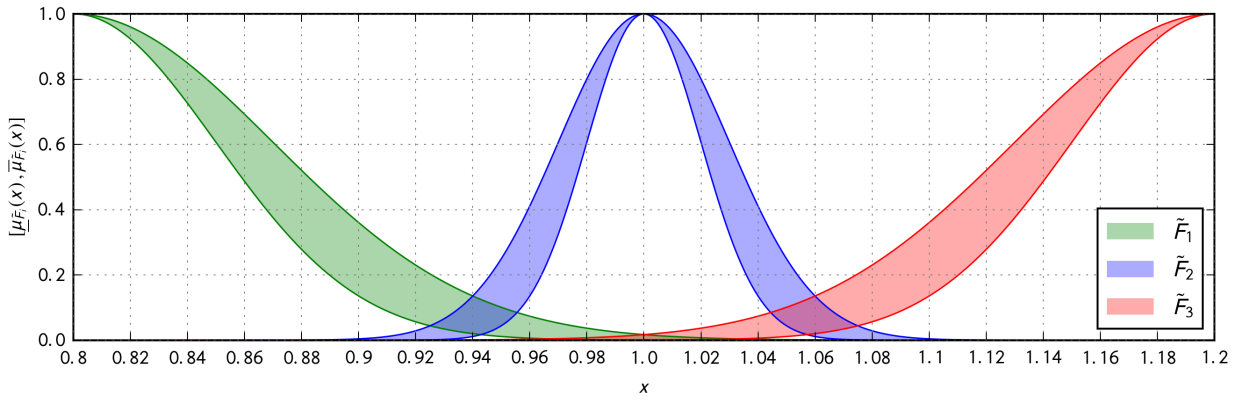

Figure 2.2: Type-2 Fuzzy Sets.

Extending this definition, a type-2 fuzzy set is denoted by $\tilde{F}$, and defined as [32, 67];

$$\tilde{F} = \int_{x \in X} \int_{u \in J_x \subseteq [0,1]} 1 \,/\, (x,u) = \int_{x \in X} \left[\int_{u \in J_x \subseteq [0,1]} 1 \,/\, u \right] \Big/ x \tag{2.2}$$

where $x \in X$ is the primary variable, $u \in J_x$ is the secondary variable, J_x is the primary membership of x, $\int \int$ denotes the union over all admissible x and u, and the secondary grades of $\tilde{F}$ are all 1. The shaded area in Figure 2.2, denoted as FOU, defines the imprecision. Thus, the FOU of $\tilde{F}$ is given by the union of all primary memberships;

$$FOU(\tilde{F}) = \bigcup_{\forall x \in X} J_x = \{(x,u) : u \in J_x \subseteq [0,1]\} \tag{2.3}$$

As J_x is an interval set defined by;

$$J_x = \left\{ (x,u) : u \in \left[\underline{\mu}_{\tilde{F}}(x), \overline{\mu}_{\tilde{F}}(x) \right] \right\} \tag{2.4}$$

and given in (2.2) and (2.3) $FOU(\tilde{F})$ can also be expressed in those terms as;

$$FOU(\tilde{F}) = \bigcup_{\forall x \in X} \left[\underline{\mu}_{\tilde{F}}(x), \overline{\mu}_{\tilde{F}}(x) \right] \tag{2.5}$$

where the upper and lower MFs $\overline{\mu}_{\tilde{F}}$, $\underline{\mu}_{\tilde{F}}$ of $\tilde{F}$ bound the FOU of $\tilde{F}$ as depicted in

Figure 2.2. Hence, (2.5) can be rewritten as,

$$\overline{\mu}_{\tilde{F}}(x) = \overline{FOU}(\tilde{F}) \; \forall x \in X \tag{2.6}$$
$$\underline{\mu}_{\tilde{F}}(x) = \underline{FOU}(\tilde{F}) \; \forall x \in X$$

where $\underline{FOU}, \overline{FOU}$ refer to the upper and lower bounds of FOU, ergo the upper and lower MFs. All these definitions can be again mapped to the previous example using Figure 2.2. Note now, that the bounded shaded area depicts the different FOUs, providing uncertainty in the shape and magnitude of the MFs. Again, the variable x represents the per unit voltage for a given node of a network. Its state is now be defined by three type-2 fuzzy sets $\tilde{F}_1$, $\tilde{F}_2$, and $\tilde{F}_3$ which are blurred versions of F_1, F_2, and F_3. . These MFs are gaussian ones with uncertain deviation and parameters (i.e. $[N(m_F, \underline{\sigma}_F; x), N(m_F, \overline{\sigma}_F; x)]$), $m_{F_1} = 0.8$, $m_{F_2} = 1.0$, $m_{F_3} = 1.2$, $\underline{\sigma}_{F_1} = 0.05$, $\underline{\sigma}_{F_2} = 0.02$, $\underline{\sigma}_{F_3} = 0.05$, and $\overline{\sigma}_{F_1} = 0.07$, $\overline{\sigma}_{F_2} = 0.03$, $\overline{\sigma}_{F_3} = 0.07$ which denote the means, lower and upper deviations respectively. Also note that, these fuzzy sets denote again if the voltage is low, acceptable or high through their interval MFs $[\underline{\mu}_{F_1}(x), \overline{\mu}_{F_1}(x)]$, $[\underline{\mu}_{F_2}(x), \overline{\mu}_{F_2}(x)]$, and $[\underline{\mu}_{F_3}(x), \overline{\mu}_{F_3}(x)]$. Hence, now for $x_n = 0.96$, the interval firing levels for the different sets are $[\underline{\mu}_{F_1}(0.96), \overline{\mu}_{F_1}(0.96)] = [0.01, 0.07]$, $[\underline{\mu}_{F_2}(0.96), \overline{\mu}_{F_2}(0.96)] = [0.14, 0.41]$, and $[\underline{\mu}_{F_3}(0.96), \overline{\mu}_{F_3}(0.96)] = [0, 0]$. These now means that when the voltage 0.96 per unit is low with $[0.01, 0.07]$ and acceptable with $[0.14, 0.41]$ degrees of truth, blurring the crisp degrees of truth from the Figure 2.1. Finally, set operations which fuzzy union, intersection and complement, are defined as follows;

$$\tilde{F}_1 \cup \tilde{F}_2 = 1 \Big/ \bigcup_{\forall x \in X} \left[\underline{\mu}_{\tilde{F}_1}(x) \vee \underline{\mu}_{\tilde{F}_2}(x), \overline{\mu}_{\tilde{F}_1}(x) \vee \overline{\mu}_{\tilde{F}_2}(x) \right] \tag{2.7}$$

$$\tilde{F}_1 \cap \tilde{F}_2 = 1 \Big/ \bigcup_{\forall x \in X} \left[\underline{\mu}_{\tilde{F}_1}(x) \star \underline{\mu}_{\tilde{F}_2}(x), \overline{\mu}_{\tilde{F}_1}(x) \star \overline{\mu}_{\tilde{F}_2}(x) \right] \tag{2.8}$$

$$\overline{\tilde{F}} = 1 \Big/ \bigcup_{\forall x \in X} \left[1 - \underline{\mu}_{\tilde{F}}(x), 1 - \overline{\mu}_{\tilde{F}}(x) \right] \tag{2.9}$$

where $\vee$, $\star$ represent t-conorm and t-norm respectively which are generalisation of the common two-valued logical functions from classical logic for fuzzy logic. Thus, $a \star b$ is a function from $[0,1][0,1]$ to $[0,1]$ that is commutative, associative, monotonic, where 1 acts as identity element. And $a \vee b$ is function from $[0,1][0,1]$ to $[0,1]$ which returns $1-(1-a \star 1-b)$. Also note that the meet and join set operations refer to intersection and union at each x, respectively. These are relevant for and/or operations.

2.3 Type-2 Fuzzy Logic Systems

FLSs [76, 77], are knowledge-based systems which use linguistic if-then rules and fuzzy sets to build non-linear mappings. Just like every other expert system [78], they select an adequate output according to their collection of rules which are the core of the system. Type-2 FLSs are FLSs which use type-2 fuzzy sets.
The basic structure of a type-2 FLS consists of a fuzzifier, a rule-base, an inference engine, a type-reducer and a defuzzifier (Figure 2.3). A type-1 FLS differs in that it does not require the type-reduction step as it uses type-1 fuzzy sets (i.e. $\underline{\mu}(x_i) = \overline{\mu}(x_i)$). Nevertheless, it is considered a special case of type-2 fuzzy sets where upper and lower MFs match. Hence, the process of mapping a given collection of inputs to a determined output is called inference, and the fuzzification and defuzzification are the transformation steps from crisp inputs to fuzzy inputs and vice-versa. The process starts with the crisp inputs being fuzzified into fuzzy sets which activate the inference process and the rule-base. Thus, fired rules are combined to produce output fuzzy sets. These are then combined and map to type-reduced sets (i.e. intervals which provide a measure of the uncertainty as it flows through the system) via a type-reducer which leads to the final defuzzification step into crisp outputs. In the type-1 case, the combined output is directly defuzzified.

2.3.1 Fuzzification

This process aims at converting crisp inputs into input fuzzy sets, a type-2 fuzzy set $\tilde{F}$ is characterised by its FOU bounded by both MFs (Figure 2.2), as explained in

Section 2.2.
As consequence, the fuzzification process inherently relies in the blurred shape of the antecedent MFs (e.g. trapezoidal, triangular or gaussian, among others). According to [79] where a shape comparison is performed, gaussian MFs need less parameters to be represented, having therefore less parameters to tune or to optimise. Also, gaussian shapes are faster for small rule-bases and provide a continuous surface that allows for gradient-based methods and also favours stability and robustness.
Hence, using for simplicity singleton fuzzification [32] and gaussian MFs with uncertain deviation as in [25, 26] (Figure 2.2). A MF $\mu_{\tilde{F}} = [\underline{\mu}_{\tilde{F}}, \overline{\mu}_{\tilde{F}}]$ for a given set F can be defined as;

$$\underline{\mu}_{\tilde{F}} = \underline{\mu}_{\tilde{F}}(x) = N(m, \underline{\sigma}; x) = e^{-\frac{1}{2}((x-m)/\underline{\sigma})^2} \tag{2.10}$$

$$\overline{\mu}_{\tilde{F}} = \overline{\mu}_{\tilde{F}}(x) = N(m, \overline{\sigma}; x) = e^{-\frac{1}{2}((x-m)/\overline{\sigma})^2} \tag{2.11}$$

2.3.2 Rule-base

If-then rules use interval type-2 fuzzy sets as rule antecedents and intervals as rule consequents. Hence, considering a system with p inputs and M rules, let the l-th rule be denoted by R_l [32] as;

$$R_l : \textbf{ if } x_1 \text{ is } \tilde{F}_{l,1} , \ldots, \text{ and } x_p \text{ is } \tilde{F}_{l,p} \textbf{ then } h_l(\mathbf{x}) \tag{2.12}$$

where $\tilde{F}_{l,i}$ represents the type-2 antecedent fuzzy sets and the consequent parameters of l-th rule, $h_l(\mathbf{x})$, can either follow a Takagi-Sugeno inference and be a multivariate linear regression (i.e. $c_{l,0} + \sum_{i=1}^{p} x_i c_{l,i}$) or follow a Mamdani inference and be a singleton that represents a type-2 fuzzy set (i.e. $[c_{l,l}, c_{l,r}]$) [32]. The use of fuzzy sets as consequents introduces a computational burden but it allows to describe mappings in more intuitive manner, therefore this approach is followed in chapter 4. On the contrary Takagi-Sugeno systems are computationally more efficient and work well in conjunction with gradient-based optimisation and adaptive techniques. Hence, used in chapter 3, for mapping the short-term load behaviours. Particularly the ones from the Class A2-C0, which indicates a special case where the antecedents are type-2 fuzzy antecedents and consequents are crisp numbers.

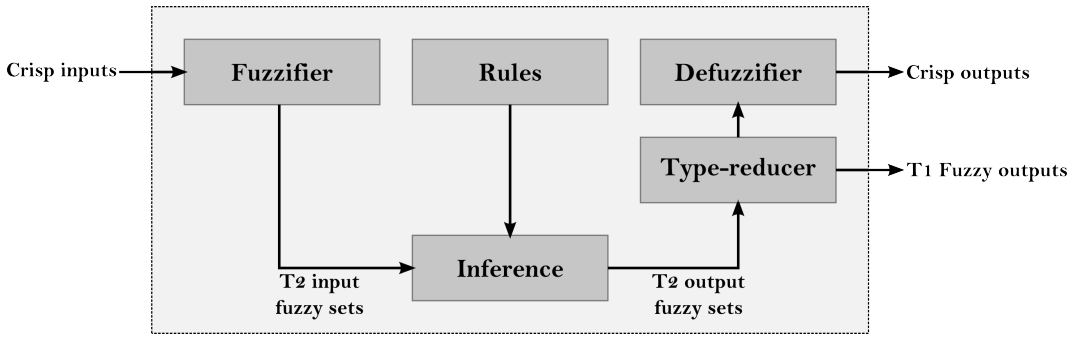

Figure 2.3: Type-2 FLS

2.3.3 Inference

Hence, with meet under product t-norm [32], given (2.8), (2.10) and (2.12) and an input set $\mathbf{x}$, the result of the input and antecedent operations for the l-th rule is an interval type-1 fuzzy set $[\underline{f}_l(\mathbf{x}), \overline{f}_l(\mathbf{x})]$, and defined as;

$$\underline{f}_l(\mathbf{x}) = \prod_{i=1}^{p} \underline{\mu}_{\tilde{F}_{l,i}}(x_i) = \prod_{i=1}^{p} N(m_{l,i}, \underline{\sigma}_{l,i}; x_i) = \prod_{i=1}^{p} e^{-\frac{1}{2}((x_i - m_{l,i})/\underline{\sigma}_{l,i})^2} \tag{2.13}$$

$$\overline{f}_l(\mathbf{x}) = \prod_{i=1}^{p} \overline{\mu}_{\tilde{F}_{l,i}}(x_i) = \prod_{i=1}^{p} N(m_{l,i}, \overline{\sigma}_{l,i}; x_i) = \prod_{i=1}^{p} e^{-\frac{1}{2}((x_i - m_{l,i})/\overline{\sigma}_{l,i})^2} \tag{2.14}$$

Therefore using the sets from Figure 2.2, for a system with inputs x_1, x_2 that represent per unit voltages at two nodes of a distribution network. the firing level of the antecent of a rule which relates both (i.e. **if** x_1 is $\tilde{F}_1$and x_2 is $\tilde{F}_2$) is the product of its interval degrees of truth and it will be use later to establish the response of the combination of rules which conform the FLS.

2.3.4 Type-Reducer & Defuzzifier

These collection of interval rule outputs have to be reduced in order to obtain the system type-reduced output, which will be defuzzified afterwards. This type-reduced set is particularly interesting as it provides a measure of the uncertainty as it flows through the system [80]. In a similar fashion as a confidence interval does it for a probabilistic system [74]. Thus, the type-reduced set grows as the uncertainty in the type-2 fuzzy sets does. The interval firing levels increase their difference and therefore its combination as well. On the contrary, the type-reduced set bounds equalise as the uncertainty in the type-2 fuzzy sets does, and therefore the interval

firing levels also tend to equalise. Hence, it is required if one seeks to cascade systems and avoid any possible information loss [59].

Hence, given that a centroid in an interval type-2 is an interval type-1 [67, 81], only the calculation of the two end points is required $[y_l, y_r]$, i.e.

$$[c_l, c_r] \equiv \int_{c_1 \in [c_{l,1}, c_{r,1}]} \cdots \int_{c_M \in [c_{l,M}, c_{r,M}]} \int_{f_1 \in [\underline{f_l}, \overline{f_1}]} \cdots \int_{f_1 \in [\underline{f_M}, \overline{f_M}]} 1 \Big/ \frac{\sum_{i=1}^{M} f_i c_i}{\sum_{i=1}^{M} f_i} \tag{2.15}$$

where for the center-of-sets, is the most common approach to obtain the type-reduced set, $[c_{l,1}, c_{r,1}]$ are pre-computed and represent the centroid of the l-th rule, $[\underline{f}_l, \overline{f}_l]$ the lower and upper firing of the l-th rule and M the number of rules.

There are several type-reduction algorithms to perform the task of obtaining $[c_l, c_r]$ [32, 81, 82, 83, 84]. For instance, computations for $[c_l, c_r]$ can be simplified by using uncertainty bounds to obtain an approximation or by using the non-closed Karnik-Mendel algorithm. The latter, widely common in the literature, calculates the bounds in an iterative fashion. Starts by sorting $\{c_{l,1}, \ldots, c_{l,1}\}$, $\{c_{r,1}, \ldots, c_{r,1}\}$ from lower to higher respectively. An initial guess of c_l, c_r is then made using the average firing of the rules $c_l = \sum_{i=1}^{M} c_{l,i}(\underline{f_i} + \overline{f_i}) / \sum_{1}^{M}(\underline{f_i} + \overline{f_i})$, $c_r = \sum_{i=1}^{M} c_{r,i}(\underline{f_i} + \overline{f_i}) / \sum_{1}^{M}(\underline{f_i} + \overline{f_i})$, to then find the switch points L and R and compute c_l, c_r i.e.

$$c_{l,L} \leq c_l \leq c_{l,L+1} \tag{2.16}$$

$$c_{l,R} \leq c_r \leq c_{l,R+1} \tag{2.17}$$

$$c_l = \frac{\sum_{i=1}^{L} c_{l,i} \overline{f}_i + \sum_{i=L+1}^{M} c_{l,i} \underline{f}_i}{\sum_{i=1}^{L} \overline{f}_i + \sum_{i=L+1}^{M} \overline{f}_i} \tag{2.18}$$

$$c_r = \frac{\sum_{i=1}^{R} c_{r,i} \underline{f}_i + \sum_{i=R+1}^{M} c_{r,i} \overline{f}_i}{\sum_{i=1}^{R} \underline{f}_i + \sum_{i=R+1}^{M} \underline{f}_i} \tag{2.19}$$

$$y = \frac{c_l + c_r}{2} \tag{2.20}$$

Then the algorithm repeats checks if the new values for c_l, c_r are the same, otherwise repeats (2.16-2.19) until c_l, c_r no longer change. Once these bounds are gathered, the type-reduced set which combines all the rules firings is obtained, and the system crisp can also be simply obtained by computing the average of c_l and c_r. The election of an average as a method of defuzzification is inspired in the common centroid

computation performed in type-1 FLSs.
Alternatively, type-reduction can be bypassed and the crisp output directly obtain from the rule firing levels and consequents, as in the Nie-Tan (2.22) [85] and NN-based algorithms [26]. These algorithms have shown a higher degree of prediction accuracy compared to other type-reduction algorithms [26]. Particularly, the Nie-Tan is broadly used as it represents a closed-form, that does not require any previous knowledge and is less computationally intensive.

$$h_i(\mathbf{x}) = c_{l,0} + \sum_{i=1}^{p} x_i c_{l,i} \tag{2.21}$$

$$y(\mathbf{x}) = \frac{\sum_{i=1}^{M} \underline{f}_i(\mathbf{x})\ h_i(\mathbf{x}) + \overline{f}_i(\mathbf{x})\ h_i(\mathbf{x})}{\sum_{i=1}^{M} \underline{f}_i(\mathbf{x}) + \overline{f}_i(\mathbf{x})} \tag{2.22}$$

2.4 Summary

Type-2 FLSs have many substantial advantages when compared to their type-1 counterparts. They provide a smoother control due to the smother shape of the control surface [71], this is due to the fact that transitions between triggered rules are less abrupt when compared to type-1 FLS because of the use of interval firings. A more adaptive system can be obtained by allowing more complex relationships which are not possible with type-1 fuzzy schemes [69]. Type-1 FLSs are merely reduced versions of their type-2 counterparts where the upper and lower MFs are the same. Also, the number of fuzzy sets used [32] can be reduced as the input range can be covered with a fewer number of sets obtaining similar input-output mappings. Furthermore, the offer an increase of the system robustness [71] due to its smoother response profile they handle better disturbances around the controlling point. It should also be noted that the controller can be thought of as a collection of different embedded controllers due to the fact, that each input or output is represented through a large number of type-1 fuzzy sets, all the ones embedded in the interval type-2 [67]. Hence, type-1 sets are just a special case of a type-2 one where there is no uncertainty, and this scheme represents a generalisation which allows for more flexible definitions. Therefore, in this case, like in many other real world applications [72], type-1 fuzzy systems may not handle the system properly.

3 Short-Term Load Forecasting in Power Systems

This chapter presents a comprehensive analysis of the design of type-2 FLSs for STLF applications. Towards improved accuracy and automating the learning process, two novel techniques are introduced in this thesis, an information-interaction feature selector and a memetic algorithm. In the former, statistically significant features are selected based on their relevancy/redundancy ratio; in the latter, the system's structure and parameters are simultaneously learnt through a problem-tailored genetic algorithm with an embedded Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm which acts as a local optimiser. Out-of-sample accuracy improvements for market-level one-day-ahead 30-minutes-resolution load forecasting are presented, and compared to other common combinations of feature selection and fuzzy learning.

3.1 Introduction

In [25], an electricity-domain AI-focused comparative study was performed, indicating that type-2 FLSs outperform type-1 FLSs and NNs in terms of generalisation power and forecasting error when using a similar number of parameters do define both systems. Nevertheless, accuracy improvements can still be gathered as the model identification is carried out in a sub-optimal manner and this fact can inherently limit their predictive performance. Hence, in type-2 FLSs, as in type-1 FLSs, model identification process is an important yet challenging problem especially but not limited to the STLF application. Model identification mainly comprises of three interwoven tasks; feature selection, structure identification and parameter determination. The first involves input selection. The second, rule generation, including input-space partitioning, MF specification, and rule-set optimisation. The third, involves fine-tuning the MFs which describe rule antecedents and consequents. However, as the number of inputs to consider increases [86] and patterns are complex like in STLF, completing these inter-related tasks in a non suitable fashion can strongly affect the prediction accuracy, as it happens with NNs [87]. Hence, if one follows a

conventional approach and performs these tasks in a sequential trial-and-error manner, model inputs have to be firstly selected. Obviously, irrelevant or redundant features can degrade the forecasting accuracy by reducing generalisation due to over-fitting, and also produce intractability issues or longer learning times when addressing the two remaining tasks (i.e. structure identification and parameter determination) due to a larger input-space. Note that, for example the selected inputs could be weather observations/predictions, calendar-based ones, or simply a subset of lags from the previous week demand which represent the candidate set [19, 31, 88]. There are two main methods to be considered when addressing feature selection in time-series forecasting; filter and wrapper approaches [9]. The former, select a subset of inputs based on its relationship with the output to predict using a metric that can be linear (e.g. Pearson correlation) [89] or non-linear (e.g. mutual information or symmetrical uncertainty) [90]. The latter, focus on building models in an exhaustive manner to use the forecast accuracy as a metric for the selection but are more prone to over-fitting issues. This means that even if they can outperform filter methods in accuracy, they are much more computationally expensive, as many type-2 FLSs have to be built. This can be impractical given that their optimal structure is also unknown and many models for different time horizons have to be built to avoid accumulative errors [91]. Thus, a key issue to address is which should be the right filtering metric for type-2 FLSs, if one seeks to increase forecasting accuracy without having to rely on a computationally expensive exhaustive-search.

Partial auto-correlation [25, 89], LASSO, elastic nets, ridge regressors [92], random forests [93], correlation-based [90] and mutual information-based [88] filter methods have been used for this feature selection task or similar ones. However, these methods are either constrained by a linear approximation to the STLF problem as short-term load behaviour could be non-linear or they lack an adequate metric that accounts for relevancy versus redundancy which can lead to over-fitting or under-fitting issues. Thus, in order to address this limitations in computationally efficient manner, a bin-less pairwise information-interaction estimator, which accounts for feature relevancy and redundancy [94] based on k-nearest neighbours sampling [95, 96] is proposed here. By means of a greedy-search which uses the Hampel distance [97], a method that selects candidate features until they stop being statistically significant can be obtained. The Hampel distance is a metric that is used as a stopping criterion to determine whether an element of a set given its score, it is

an outlier or not. Thus, it represents a robust version of the $3-\sigma$ approach to outlier detection. A candidate feature is considered significant, if according to the Hampel distance is an outlier within the set of candidates features. All according to their predictive power and/or shared un-/conditional mutual information.

Once the model inputs are selected, structure identification and parameter determination can be tackled. Firstly, as it can be considered that each rule defines a load pattern from a local perspective, clustering techniques that do not require to specify the number of clusters in the data apriori (e.g. Subtractive clustering [98, 99], or DBSCAN [100, 101]) can be used here for the structure identification task. Then, a numerical optimisation (e.g. evolutionary algorithms [25] , or gradient-based methods [32]) for the previously defined structure has to be carried out. This leads to a costly a trial-and-error loop if one seeks to improve accuracy, especially if this also involves also repeating a parametric feature selection process. However, despite the fact that the model structure cannot be known upfront, both tasks can be carried out simultaneously using variable-length memetic algorithms [102, 103, 104, 105]. In this case, a genetic algorithm with tailored individuals' structure [106], operators and the BFGS [107, 108], or any other local-search procedure embedded in it. An important advantage of the BFGS quasi-Newton method is that uses a Hessian approximation only using the first derivative, which already are complex to gather. Besides, it is a robust algorithm with super-linear convergence for solving large non-linear optimisation problems like this one.

Therefore, this chapter makes two distinctive contributions to the design type-2 FLSs for STLF applications. Firstly, it introduces the aforementioned information-interaction feature selection procedure. Secondly, it proposes the use of a variable-length memetic learning mechanism to concurrently learn structure and parameters of type-2 FLSs. Both techniques and their combination, are novel within the power system data-driven domain, they allow for finding a predictor which accurately models the data independently from any kind of initial assumption in a time-effective fashion. Furthermore, accuracy improvements are also obtained as Section 3.4 will illustrate. Hence, this chapter brings to light the type-2 FLSs generalisation power for STLF applications.

The rest of the chapter is organised as follows: Section 3.2 and 3.3 describe the chapter contributions including their different steps, the proposed information-interaction feature selector and the variable-length memetic learning

scheme for their definition. In Section 3.4, both contributions are tested and evaluated against other common approaches in literature on chaotic systems and load data-sets from the Australian electricity market. Finally, conclusions and future work are drawn in Section 3.5.

3.2 Information-Interaction Feature Selection (Algorithm 1)

Filter methods for feature selection play an essential role in type-2 FLSs for STLF. From a set of candidate inputs such as historical demand, type-of-day or meteorological conditions, a representative subset is selected for the forecasting task. As mentioned, this task is commonly performed using partial auto-correlation [25, 89], LASSO, elastic nets, ridge regressors [92], random forests [93], correlation-based [90] and mutual information-based [97, 88] filter methods, as discussed in the introduction. The former rely on the identification of linear dependencies whilst the two last ones can identify linear and non-linear dependencies. However, if the linearity assumption between STLF inputs does not hold or an inappropriate metric for redundancy versus relevancy is not used, the selected inputs can constrain the accuracy of the produced forecast.

Hence, the use of an information-interaction [94] metric is proposed here to avoid both limiting factors. The information-interaction metric accounts for features' relevancy and any un-/conditional redundancies by relying in on the definition of the differential entropy (3.1).

$$H(X) = -\int f(x) log(f(x)) dx \tag{3.1}$$

where X is a random variable with an unknown density function $f(x)$. It provides an uncertainty measurement, whilst higher order entropies like mutual information $H(X_i; X_j)$ and conditional mutual information $H(X_i; X_j \mid Y)$, quantify the amount of information shared between the variables X_i, X_j given a third one Y. Using these, the first-order information-interaction between a new feature X_i, a set of already selected features S, and the output to be predicted Y can be defined as (3.2) [94].

$$I_{X_i,S,Y} = H(X_i; Y) - \beta \sum_{X_j \in S} H(X_i; X_j) + \gamma \sum_{X_j \in S} H(X_i; X_j \mid Y) \tag{3.2}$$

Therefore, by using (3.2), the utility of a new feature is defined by a trade-off between its shared information with the output to be predicted and its un-/conditional correlations with the selected features (i.e. feature relevancy versus feature redundancy). Positive values imply that the information shared by S and Y has been increased as result of including X_i. The positive parameters β and γ allow for the parametrisation of these relevancy/redundancy penalties. High order entropies in (3.2) (i.e. $H(X_i; X_j)$ and $H(X_i; X_j \mid Y)$) can be estimated without directly binning the data and estimating the probability distribution through non-parametric bin-less estimators [95, 109]. The principle behind them is to average local entropy contributions in the neighbourhood of each point by using the point k-nearest neighbours. The strength of this bin-less estimation arises from its less difficult to compute local entropy rather than estimate a probability density [96]. Hence, mutual information and conditional mutual information for 1-dimensional variables which represent candidate inputs X or the output to be predicted Y can then easily be estimated independently through their time-series with N samples as (3.3), (3.4).

$$\hat{H}_k(X_i; Y) = \psi(N) + \psi(k) \; - \frac{1}{N} \sum_{i=1}^{N} (\psi(\mathrm{n}_{x_i}(n) + 1) + \psi(\mathrm{n}_y(n) + 1)) \tag{3.3}$$

$$\hat{H}_k(X_i; X_j \mid Y)) = \psi(k) \; - \frac{1}{N} \sum_{n=1}^{N} (\psi(\mathrm{n}_{x_i x_j}(n) + 1) + \psi(\mathrm{n}_{x_i y}(n) + 1) - \psi(\mathrm{n}_y(n) + 1)) \tag{3.4}$$

where ψ is the digamma function. The estimation process starts by defining a random variable ϵ_k which represents the maximum norm in all dimensions for the the k-th nearest neighbour. Then by computing the number of points (i.e. n_{x_i}, n_y, $\mathrm{n}_{x_i x_j}$, $\mathrm{n}_{x_i y}$) within ϵ_k for each sample i when one projects onto the different sub-spaces or joint spaces. Hence, (3.3), (3.4) only rely on distances between samples, and not on the space dimension [96].

Hence, once these estimators have been established (3.3, 3.4), the feature selection procedure can be simply defined as a binary search tree where in each iteration, one selects and evaluates inputs according to the information interaction metric (3.2). This combinatorial search is performed via a greedy algorithm with a stopping criterion that establishes the cardinality of the desired input subset S. Note that an exhaustive search will become intractable once the cardinality of the candidates set X increases.

Algorithm 1 Information-Interaction Feature Selector

1: **Initialisation**
Set desired input subset $S = \{\}$, and the candidates set X with all the eligible inputs.
2: **Input-Output mutual information computation** (3.3)
Estimate the mutual information $\hat{H}_k(X_i; Y)$ between each candidate input $X_i \in X$ the output Y.
3: **Initial input selection**
Find $X^\star \in X$ with the largest $\hat{H}_k(X_i; Y)$, set $S = \{X^\star\}$ and $X = X \setminus \{X^\star\}$
4: **while** $X \neq \emptyset$ **do**
5: **Input-Input mutual information computation** (3.3)
For all pairs of inputs X_i, X_j with $X_i \in X$ and $X_j \in S$ estimate $\hat{H}_k(X_i, X_j)$ if it is not already available.
6: **Input-Input conditional mutual information computation** (3.4)
For all pairs of inputs X_i, X_j with $X_i \in X$ and $X_j \in S$ estimate $\hat{H}_k(X_i, X_j \,|\, Y)$ if it is not already available.
7: **Input selection** (3.2)
Find $X^+ = \arg\max_{X_i \in X} \{I_{X_i,S,Y}\}$
8: **Stopping criterion** (3.6)
If $Z_{X^+} > 3$, then set $S = S \cup \{X^+\}$ and $X = X \setminus \{X^+\}$. Otherwise end search.
9: **return** S

Hence, given that the cardinality of the subset of relevant features is unknown, as in [97], a Z-test outlier detection method based on the Hampel distance (3.5) is used as a robust version of the 3σ approach to outlier detection and in this case as a robust stopping criteria (Step 8 in Algorithm 1). As previously mentioned, a candidate feature is considered significant, if according to the Hampel distance is an outlier within the set of candidates features. All according to their estimated predictive power and/or estimated shared un-/conditional mutual information.

$$d_{X_i} = \left| I_{X_i,S,Y} - I^{(50)}_{X_i,S,Y} \right| \tag{3.5}$$

where d_{X_i} is the absolute deviation of the information-interaction $I_{X_i,S,Y}$ of X_i and $I^{(50)}_{X_i,S,Y}$ denotes the median of the information-interaction of all $X_i \in X$. Hence, the Z-score of the new most suitable candidate X^+ can now be defined as (3.6).

$$Z_{X^+} = \frac{d_{X^+}}{1.4826\, d^{(50)}_X} \tag{3.6}$$

where $d^{(50)}_X$ is the median of the absolute distances given by (3.5) for every $X_i \in X$. Hence, by using this procedure and following the rule $Z_{X^+} > 3$, the statistical significance of X^+ can be assessed. If X^+ is an outlier, it will be added to S (i.e. so

selected variables are always safe), and removed from the candidates subset X and the search procedure will iteratively proceed until X is empty. Otherwise, the search procedure will stop not including the last X^+ and providing the subset S where all its features are outliers according to their Z-score, which relies on (3.2). That is to say, the obtained subset S contains all the inputs which according to the proposed information interaction metric are representative and significantly different from the rest of the candidate features.

3.3 Memetic Type-2 Fuzzy Logic Systems (Algorithm 2)

Traditionally, once all the necessary features are selected, Takagi-Sugeno FLSs are designed in a sequential fashion. Firstly, the data-set is usually divided according to a clustering technique, into rule antecedents defining the system structure, sometimes followed by a least-squares procedure which is used to infer the rule consequents. Followed by, a numerical optimisation, which generally is a genetic algorithm and/or a batch gradient descent, is carried out to fine-tune all the system parameters. Finally, once results are obtained, the structural initial assumptions are exhaustively modified to analyse whether improvements can be obtained. Therefore, not knowing upfront a near-optimal structure in terms of accuracy (i.e. how many fuzzy rules are required or which position or shape should the fuzzy clauses that compose those have), inherently implies modifying fixed-coded structures (i.e. with a fixed number of design parameters) which can represent a limiting factor for the technique's potential. Therefore, as introduced in Section 3.1, the goal when learning type-2 FLSs is to concurrently optimise the interwoven structure and parameters, which creates a need for a variable-coding scheme and the use of memetic algorithms which are evolutionary ones with and embedded local optimisation. Note that subsections 3.3.1 to 3.3.5, correspond to the structure and the different steps of Algorithm 2.

3.3.1 Structure of individuals

When one seeks to concurrently learn structure and parameters, the interwoven structure learning and parameter estimation, the need for a variable-coding scheme arises. Messy genetic algorithms enable this option. Messy genetic algorithms, unlike conventional genetic algorithms, do allow for flexible coding where individuals are under-/over-specified with respect to the fuzzy logic system definition [106]. Following

Algorithm 2 Memetic Type-2 FLS

1: **Initialisation**
Set algorithm's parameters: population size s, number of generations i_{max}, genetic operator probabilities pr_{cs}, pr_{cr}, pr_m and learning rate η. Randomly define the initial population with $\{A_1, \ldots, A_s\}$ individuals with random numbers of rules.
2: **Fitness computation**
Each individual A_j statistical information is computed based on (3.7) for a given training data-set D.
3: **for** $i \leftarrow 0, i_{max}$ **do**
4: **Parent selection**
Individuals are randomly paired, every individual of current population becomes a parent.
5: **Genetic operations**
Each pair of parents $\{A_1, A_2\}$ are recombined with probabilities pr_{cs}, pr_{cr} and the two obtained children $\{A'_1, A'_2\}$ are then mutated with probability pr_m respectively.
6: **Local optimisation**
Each child A'_j is optimised via a n-epochs BGFS quasi-Newton procedure for a given training data-set D.
7: **Fitness computation**
Each child A'_j statistical information is computed based on (3.7) for a given training dataset D.
8: **Deterministic tournaments**
Each child competes with one of its parents in order to be included in the next generation. The individual with best goodness-of-fit of each tournament, which is defined based on the individuals' similarity proceeds to the next generation.
9: **Rank population**
10: **return** A_{best}

a similar approach, a population of type-2 FLSs will increasingly evolve to more complex forms (i.e. the number of rules dynamically increases until optimal generalisation is achieved). Fuzzy clauses are the key elements of the coding scheme. However, contrary to [106] antecedents and consequents are divided. Type-2 fuzzy antecedents are represented by their MFs with 4-tuples like $(i, m_{l,i}, \overline{\sigma}_{l,i}, \underline{\sigma}_{l,i})$, in which i is the input index, $m_{l,i}, \overline{\sigma}_{l,i}, \underline{\sigma}_{l,i}$ the mean, upper and lower deviation of an type-2 gaussian MF with uncertain deviation from the l-th rule. Consequent parameters are represented by tuples $(i, c_{l,i})$ where i represents the input index and $c_{l,i}$ its coefficient. In the following text an example of a messy coded rule with 2 inputs is provided where two collections of unsorted 4-tuples and tuples which represent all the indices and parameters of its constituting antecedents and consequents respectively. In this example rule consequent parameters define a multivariate regression. Hence, by using this coding scheme individuals with different fuzzy sets and/or different inputs can be used and therefore with different complexities.

$$\left\{ \begin{array}{l} \{(0, c_{l,0}), (2, c_{l,2}), (1, c_{l,1})\} \\ \{(1, m_{l,1}, \overline{\sigma}_{l,1}, \underline{\sigma}_{l,1}), (2, m_{l,2}, \overline{\sigma}_{l,2}, \underline{\sigma}_{l,2})\} \end{array} \right\}$$

$$\downarrow$$

$$\underline{\mu}_{\tilde{F}_{l,1}} = N(m_{l,1}, \underline{\sigma}_{l,1}), \overline{\mu}_{\tilde{F}_{l,1}} = N(m_{l,1}, \overline{\sigma}_{l,1}), \underline{\mu}_{\tilde{F}_{l,2}} = N(m_{l,2}, \underline{\sigma}_{l,2}), \overline{\mu}_{\tilde{F}_{l,2}} = N(m_{l,2}, \overline{\sigma}_{l,2}),$$

$$h_l = c_{l,0} + x_1 c_{l,1} + x_2 c_{l,2}$$

$$\downarrow$$

$$R_l : \textbf{ if } x_1 \text{ is } \tilde{F}_{l,1} \text{ and } x_2 \text{ is } \tilde{F}_{l,2} \textbf{ then } h_l(\mathbf{x}) = c_{l,0} + \sum_{i=1}^{2} x_i c_{l,i}$$

Hence, individuals in the genetic algorithm representing type-2 FLSs can be simply defined by an unsorted and variable number of rules like the one from the previous example,

$$\{R_1, \ldots, R_l, \ldots, R_M\}$$

3.3.2 Fitness computation (Steps 2 & 7)

The learning problem can be easily formulated as an optimisation one where the main objective is to minimise the approximation error whilst avoiding over-fitting issues (i.e. lack of generality). Thus, as in statistical modelling, one of the most important issues is to find the right trade-off between the goodness-of-fit and the model complexity [65]. Following the Ockham's razor principle (i.e. among competing hypotheses, the one with the fewest assumptions should be selected) as also by limiting the number of fuzzy rules will help to prevent over-fitting. This is due to fact that the generalisation ability decreases with the number of rules (i.e. loses the capacity to generalise new data). However, if it is done too aggressively the model will be unable to produce accurate predictions, as it is desired to obtain a model that can cover the whole training data-set D with sufficient antecedents. Several statistical information criteria have been formulated [65], commonly under a general framework (3.7) which allows for comparing fuzzy models with different complexities during an evolutionary procedure.

$$\mathcal{P}(A_j, D) = log\left(\frac{1}{N}\sum_{k=1}^{N}(\hat{y}_{A_j}(\mathbf{x}^k) - y^k)^2\right) + \frac{\alpha}{N} M_{A_j} \tag{3.7}$$

where A_j is the analysed individual (i.e. a type-2 FLS) from the population of solutions, D the training data-set with N examples like $(\mathbf{x}^k : y^k)$, $\hat{y}^k_{A_j}$ the model prediction, y^k the real value for the example k, M_{A_j} a complexity metric (e.g. the number of rules of the individual A_j) and α a positive parameter that forces the second term to tend to zero as the number of examples increases (Algorithm 2: Steps 2 & 7). Thus, α is therefore the key factor for avoiding an uncontrolled growth of the individuals size as it establishes the relative rule cost. Hence, according to this criteria, the best type-2 FLS will be the one that better fits the training data using the minimum number of rules.

3.3.3 Parent Selection & Deterministic Tournaments (Steps 4 & 8)

The multi-modal nature of concurrent structure identification and parameter estimation (i.e. explore solutions with variable numbers of rules) makes this optimisation especially challenging as different minima can be found depending on the rule number. Therefore, a niching procedure denoted deterministic crowding [110, 111] is used here to reduce abrupt generational changes in the population reducing the chances of premature convergence. Thus, parents are randomly grouped in pairs and when their offspring are obtained, each one competes against one of its parents. The winner is preserved for the following generation. The matching is decided according to the individuals similarity and each winner is based on (3.7). Hence, in order to preserve the diversity within the solutions to be explored by the memetic algorithm, it is necessary to establish when two individuals are similar. However, one cannot rely in the direct comparison of their parameters (i.e. genotypes) as individuals can be composed by different number of rules. To avoid this issue, one can assume that two individuals are as similar as their produced forecasts (i.e. phenotypes). Thus, rather than computing the distance between parameters or perform set operations the distance between produced forecasts is computed. Given a sub-sample D' of the training data-set with N' input-output pairs the similarity of two individuals is defined by (3.8).

$$S(A_1, A_2) = \frac{1}{N'} \sum_{k=1}^{N'} \left| \hat{y}_{A_1}(\mathbf{x}^k) - \hat{y}_{A_2}(\mathbf{x}^k) \right| \tag{3.8}$$

Thus, by using this distance-based similarity, two pairs of parent-offspring are obtained. Only the fittest proceed to the next generation independently from the fact that they are offspring or not.

3.3.4 Genetic Operations (Step 5)

Crossover and mutation operators are here tailored to enhance the algorithm flexibility and to un-trap the local search procedure if it reaches a local minimum. A cut & splice operator as in [106] which acts on two levels is used here. On the rule-set level, to maximise the exploration, two children rule-sets are originated from two parents rule-sets not necessarily using the same crossover points. This allows for obtaining children with different numbers of rules. The following is an example of this rule-set level cut & splice operation, where two parents A_1 and A_2 are split producing two sets of rules each one with different cardinalities. Then, these are distributed among the two children A'_1 and A'_2.

$$A_1 = \{R_{1,1}, R_{1,2}, \mid R_{1,3}\}, A_2 = \{R_{2,1}, \mid R_{2,2}, R_{2,3}\}$$

$$\downarrow$$

$$A_{11} = \{R_{1,1}, R_{1,2}\}, A_{12} = \{R_{1,3}\}$$

$$A_{21} = \{R_{2,1}\}, A_{22} = \{R_{2,2}, R_{2,3}\}$$

$$\downarrow$$

Randomly distributed

$$\downarrow$$

$$A'_1 = \{R_{1,1}, R_{1,2}, R_{2,2}, R_{2,3}\}, Z'_2 = \{R_{1,3}, R_{2,1}\}$$

However, it may occur that meaningful rules from either one or both of the parents could be lost. There is no need to complete the children as cutting points are forced to be near the extremes and the local optimisation will repair them. However, despite the fact that conflicting or redundant rules will be penalised by the fitness function (3.7), it

is required to eliminate them from the children. To compensate for their deletion, random rules are added given a 0.5 probability. This not only enhances the diversity of solutions within the population, but also helps controlling the population complexity. However, as in Section 3.3.3, a direct comparison of rule parameters (i.e. genotypes) is not appropriate. It can occur that rule antecedents covering the same set of samples N' are establish as dissimilar. Thus, one can again rely again on the distances between their produced outputs (i.e. phenotypes) (3.9).

$$S(R_1, R_2) = \frac{1}{N'} \sum_{k=1}^{N'} \left| \overline{f}_1(\mathbf{x}^k) + \underline{f}_1(\mathbf{x}^k) - \overline{f}_2(\mathbf{x}^k) - \underline{f}_2(\mathbf{x}^k) \right| \tag{3.9}$$

On the rule level, alike [106], an antecedent/consequent from a randomly picked rule from the first parent is selected. The second antecedent/consequent must not be a clone rule, but it must be triggered by similar conditions. Hence, each candidate rule similarity from the second parent is assessed using (3.9) and the closest non-cloned rule is selected. Also, two children are generated again. Note that over-specification and under-specification of certain inputs can occur. For the former, a first-come-first-served approach is employed and for the latter a random completion procedure. The following is an example of this rule-level crossover operation for rule antecedents. This procedure is analogous for the rule consequents.

$$R_1 = \{\tilde{F}_{1,1}, \tilde{F}_{1,2}, \tilde{F}_{1,3}, \mid \tilde{F}_{1,4}\}, R_2 = \{\tilde{F}_{2,1}, \mid \tilde{F}_{2,2}, \tilde{F}_{2,3}, \tilde{F}_{2,4}\}$$

$$\downarrow$$

$$R_{11} = \{\tilde{F}_{1,1}, \tilde{F}_{1,2}, \tilde{F}_{1,3}\}, R_{12} = \{\tilde{F}_{1,4}\}$$

$$R_{21} = \{\tilde{F}_{2,1}\}, R_{22} = \{\tilde{F}_{2,2}, \tilde{F}_{2,3}, \tilde{F}_{2,4}\}$$

$$\downarrow$$

Randomly distributed & completed, if appropriate

$$\downarrow$$

$$R'_1 = \{\tilde{F}_{1,4}, \tilde{F}_{2,1}, \tilde{F}_{*,2}, \tilde{F}_{*,3}\}$$

$$R'_2 = \{\tilde{F}_{1,1}, \tilde{F}_{1,2}, \tilde{F}_{1,3}, \tilde{F}_{2,2}, \tilde{F}_{2,3}, \tilde{F}_{2,4}\}$$

On the other hand, in this messy-inspired coding scheme mutation, also occurs in

different levels and is tailored to the fuzzy modelling needs, for rule-sets, rules and parameters. In the first ones, a rule or a clause is either randomly removed or generated based on a randomly selected example. Similarly, in the latter, the parameters of a randomly selected antecedent or consequent are slightly modified. Note that all these crossover and mutation operations occur according to predefined probabilities for rule-set crossover pr_{cs}, rule crossover pr_{cr} and mutation pr_m respectively.

3.3.5 Local Optimisation (Steps 6)

At each iteration of the population-based evolutionary global search, a local search is carried out to unveil all models accuracy potential and speed up the learning the global search process. Embedding this local search procedure in the global evolutionary one, is what makes the proposed methodology belong to the memetic algorithms family. These methods model natural systems that combine lifetime learning and evolutionary adaptation [102, 104, 105, 112, 113].
Knowing upfront the model derivatives towards the mean squared error will be minimised represents an advantage in terms of convergence speed against derivative-free heuristics, where the steps are taken following a random direction. Thus, a local optimisation using a n-iterations fine-tuning via a BFGS [107, 108] quasi-Newton method is applied to each generation offspring, which is outlined in Algorithm 3. BFGS is an iterative algorithm procedure generally used for solving the non-linear numerical unconstrained optimisation which matches with the task of defining the right parameters for a type-2 FLS. Although, suited to the task it appears no other work has made used for this particular task.
Below are defined the necessary equations for the jacobian computation necessary to speed up the BFGS algorithm. It should be borne in mind that mean squared error is used as a performance metric and type-2 MFs with uncertain deviation, Nie-Tan type-reduction (2.22) and product as t-norm are used here. Hence, if $w_{A_j,l,i}(n)$ is an antecedent parameter (i.e. mean $m_{A_j,l,i}$, upper spread $\overline{\sigma}_{A_j,l,i}$ or lower spread $\underline{\sigma}_{A_j,l,i}$), the partial derivatives for the i-th antecedent parameter of the l-th rule given a k data pair $(\mathbf{x}^k : y^k)$ are defined in (3.10)-(3.17).

Algorithm 3 Broyden-Fletcher-Goldfarb-Shanno [107]

1: **Initialisation**
Define initial guess $\mathbf{Z}_0$ from the given offspring solution Z, define approximate Hessian matrix $B_0 \leftarrow I$ and set the rest of the algorithm parameters: maximum number of iterations i_{max} and maximum convergence tolerance τ_{max}.
2: **while** $i < i_{max}$ **and** $||\nabla f(\mathbf{x}_i)|| > \tau_{max}$ **do**
3: **Get the search direction**
$\mathbf{d}_i \leftarrow -B_i \nabla f(\mathbf{x}_i)$.
4: **Calculate step size** λ_i **via a line-search given** $\mathbf{d}_k$
5: **Update solution**
$\mathbf{Z}_{i+1} \leftarrow \mathbf{Z}_i + \lambda_i \mathbf{d}_k$
6: **Update the approximate Hessian**
$\mathbf{p}_i \leftarrow \lambda_i \mathbf{d}_i$
$\mathbf{q}_i \leftarrow \nabla f(\mathbf{x}_{i+1}) - \nabla f(\mathbf{x}_i)$
$B_{i+1} \leftarrow B_i + \dfrac{\mathbf{q}_i \mathbf{q}_i^\mathsf{T}}{\mathbf{q}_i^\mathsf{T} \mathbf{p}_i} - \dfrac{B_i \mathbf{p}_i \mathbf{p}_i^\mathsf{T} B_i}{\mathbf{p}_i^\mathsf{T} B_i \mathbf{p}_i}$
7: **return** Z

$$\frac{\partial e^k_{A_j}}{\partial w_{A_j,l,i}}\Big|_k = \frac{e^k_{A_j}(h_l(\mathbf{x}^k) - \hat{y}^k)}{\sum_{i=1}^{M} \overline{f}_i(\mathbf{x}^k) + \underline{f}_i(\mathbf{x}^k)} \times \left(\frac{\partial \overline{f}_l(\mathbf{x}^k)}{\partial w_{A_j,l,i}}\Big|_k + \frac{\partial \underline{f}_l(\mathbf{x}^k)}{\partial w_{A_j,l,i}}\Big|_k \right) \tag{3.10}$$

$$\frac{\partial \overline{f}_l(\mathbf{x}^k)}{\partial w_{A_j,l,i}}\Big|_k = \frac{\partial \mu_{\underline{\tilde{F}}_{l,i}}}{\partial w_{A_j,l,i}}\Big|_k \prod_{\substack{j=1 \\ j\neq i}}^{p} N(m_{l,j}, \overline{\sigma}_{l,j}; x^k_j) \tag{3.11}$$

$$\frac{\partial \underline{f}_l(\mathbf{x}^k)}{\partial w_{A_j,l,i}}\Big|_k = \frac{\partial \mu_{\underline{\tilde{F}}_{l,i}}}{\partial w_{A_j,l,i}}\Big|_k \prod_{\substack{j=1 \\ j\neq i}}^{p} N(m_{l,j}, \underline{\sigma}_{l,j}; x^k_j) \tag{3.12}$$

$$\frac{\partial \overline{\mu}_{\tilde{F}_{l,i}}}{\partial m_{A_j,l,i}}\Big|_k = \frac{(m_{l,i} - x^k_i)}{(\overline{\sigma}_{l,i})^2} N(m_{l,i}, \overline{\sigma}_{l,i}; x^k_i) \tag{3.13}$$

$$\frac{\partial \underline{\mu}_{\tilde{F}_{l,i}}}{\partial m_{A_j,l,i}}\Big|_k = \frac{(m_{l,i} - x^k_i)}{(\underline{\sigma}_{l,i})^2} N(m_{l,i}, \underline{\sigma}_{l,i}; x^k_i) \tag{3.14}$$

$$\frac{\partial \overline{\mu}_{\tilde{F}_{l,i}}}{\partial \overline{\sigma}_{A_j,l,i}}\Big|_k = \frac{(m_{l,i} - x^k_i)^2}{(\overline{\sigma}_{l,i})^3} N(m_{l,i}, \overline{\sigma}_{l,i}; x^k_i) \tag{3.15}$$

$$\frac{\partial \underline{\mu}_{\tilde{F}_{l,i}}}{\partial \underline{\sigma}_{A_j,l,i}}\Big|_k = \frac{(m_{l,i} - x^k_i)^2}{(\underline{\sigma}_{l,i})^3} N(m_{l,i}, \underline{\sigma}_{l,i}; x^k_i) \tag{3.16}$$

$$\frac{\partial \underline{\mu}_{\tilde{F}_{l,i}}}{\partial \overline{\sigma}_{A_j,l,i}}\Big|_k = \frac{\partial \overline{\mu}_{\tilde{F}_{l,i}}}{\partial \underline{\sigma}_{A_j,l,i}}\Big|_k = 0 \tag{3.17}$$

On contrary, if $w_{A_j,l,i}$ is a linear consequent parameter (i.e. $c_{A_j,l,i}$), the partial derivatives are shown in (3.18).

$$\frac{\partial e^k_{A_j}}{\partial c_{A_j,l,i}}\Big|_k = \begin{cases} \dfrac{e^k_{A_j}(\overline{f}_l(\mathbf{x}^k) + \underline{f}_l(\mathbf{x}^k))}{\sum_{i=1}^{M} \overline{f}_i(\mathbf{x}^k) + \underline{f}_i(\mathbf{x}^k)}(x_i^k) & \text{if } i \neq 0 \\ \dfrac{e^k_{A_j}(\overline{f}_l(\mathbf{x}^k) + \underline{f}_l(\mathbf{x}^k))}{\sum_{i=1}^{M} \overline{f}_i(\mathbf{x}^k) + \underline{f}_i(\mathbf{x}^k)} & \text{otherwise} \end{cases} \tag{3.18}$$

3.4 Experimental Setup, Data & Results

k	β	γ	$size$	i_{max}	$epochs$	α	$pr_{cs/cr/m}$
3	0.75	0.25	30	100	10	10	0.6, 0.1, 0.05

Table 3.1: Design Parameters for Algorithms 1 & 2

3.4.1 Chaotic Systems

Chaotic behaviour can be described as bounded fluctuations of the output of a non-linear system with high degree of sensitivity to initial conditions [114]. That is to say, trajectories with nearly identical conditions can differ a lot from each other. The Mackey-Glass equation (for $\tau > 17$ exhibits chaos) is a non-linear delay differential equation that has become one of the main benchmarks for AI-based time-series prediction [32].

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1 + x(t-\tau)^c} - bx(t) \tag{3.19}$$

Chosen constants are $\tau = 17$, $a = 0.2$, $b = 0.1$ and $c = 10$ as in [115]. Hence, the objective is to assess if the proposed learning method is able to produce an type-2 FLS which accurately model x_{k+6} from a sample of 2000 input-output data pairs, the first 1000 pairs for training and the remaining ones for testing as shown in Fig. 3.1. Selected set of inputs are $\{x_{k-18}, x_{k-12}, x_{k-6}, x_k\}$ (Fig. 3.2). Table 3.1 depicts the design parameters, these remain fixed during both experiments. The key for their selection is achieving a balance between the selection threshold, weighting factors and cut & splices rates to obtain a slow growth of the population complexity. Hence, a comparison with a NN and type-2 FLSs defined via different structure and parameter learning combinations was then performed. NN parameters are defined via mean squared error cross-validation procedure using the training dataset. To determine the

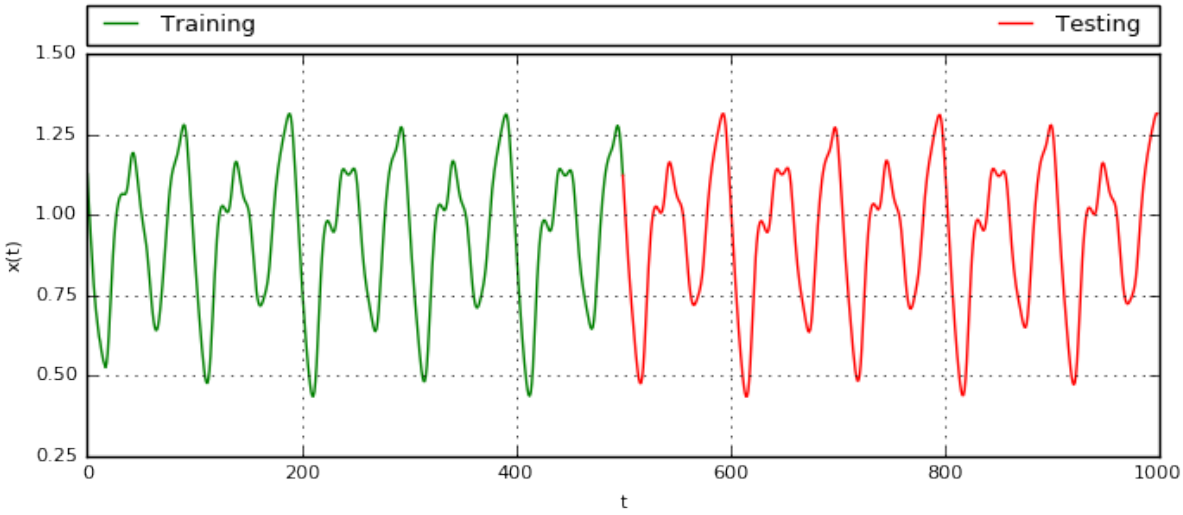

Figure 3.1: Mackey-Glass training & testing datasets.

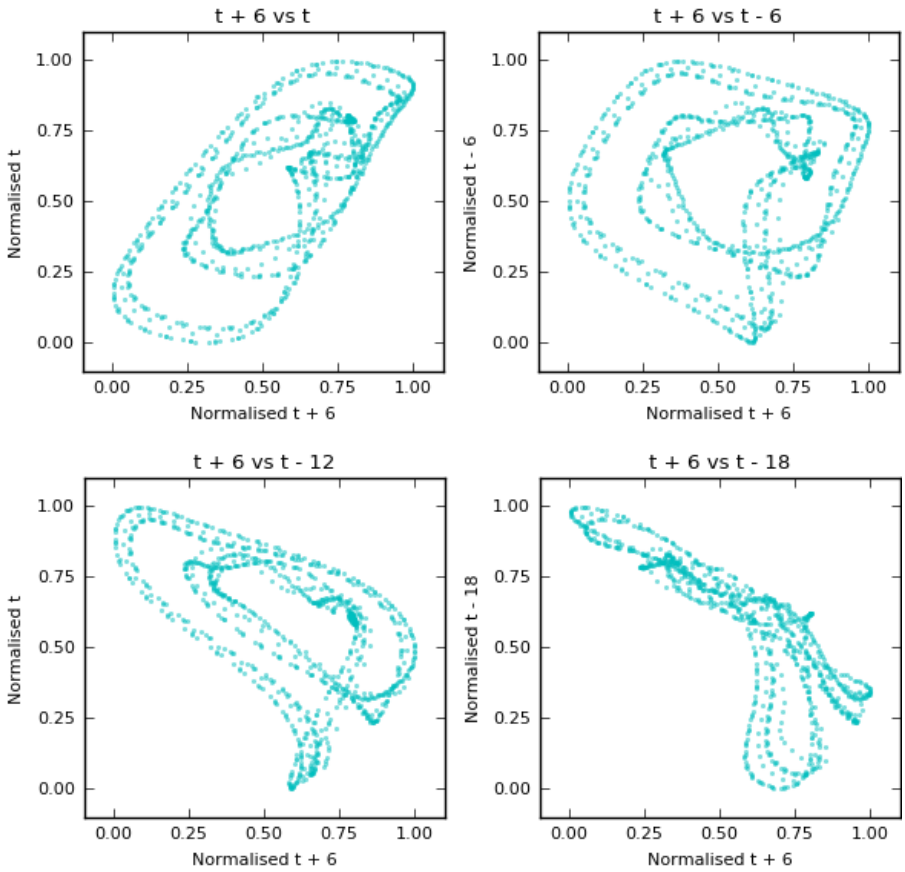

Figure 3.2: Mackey-Glass output versus input scattergrams.

Method	Structure & Parameters	MAPE
NN	Cross-Validation & BFGS	1.546 %
Type-2 FLS	DBSCAN & BGFS	0.405 %
	DBSCAN & Genetic Algorithm	0.391 %
	Subtractive Clustering & BGFS	0.302 %
	Subtractive Clustering & Genetic Algorithm	0.277 %
	Memetic Algorithm	**0.226** %

Table 3.2: Mackey-Glass out-of-sample results

structure, subtractive clustering and DBSCAN were used for this tasks, each time contrary to the memetic case, their parameters were established through cross-validation procedure using silhouette [116] as a metric. Although impractical in real applications due to its computational effort, it represents the best-case scenario for both. Finally, to establish the FLS parameters, a genetic algorithm and BFGS were used, which represent a global and local heuristic, respectively. All to explore the initial suitability of the approach to the STLF problem. Results can be found in Table 3.2 where the mean average percentage error (MAPE) over 10 experiments is used as performance index. These results depict the competence of the proposed scheme (Algorithm 2) to carry out time-series forecasting (Fig. 3.3). Improvements over the NN and other fuzzy learning schemes are mainly due to the use of a more competent model structure and parameters to generalise (Table 3.2 & Figure 3.3).

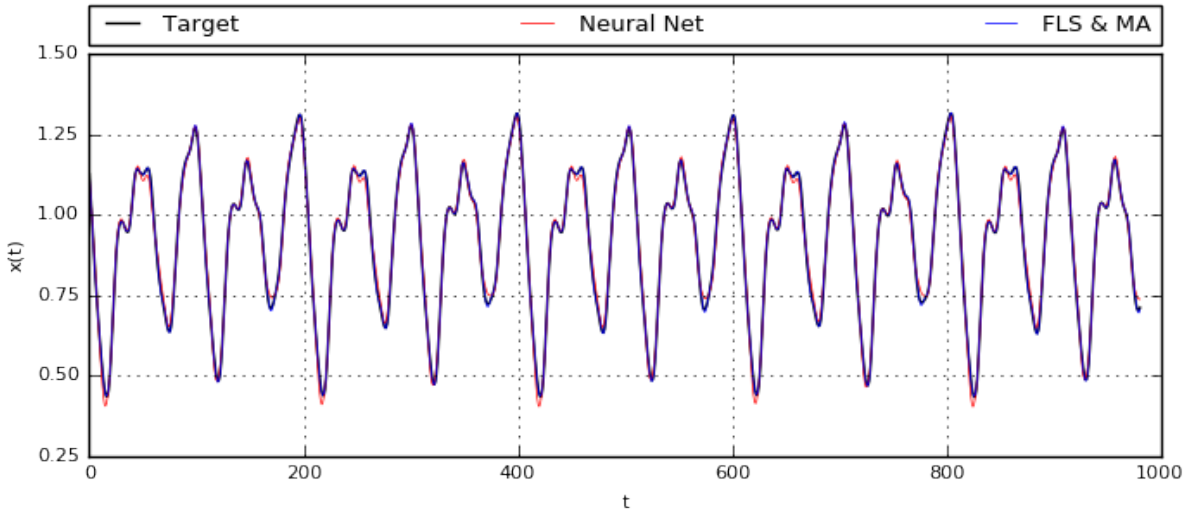

Figure 3.3: Mackey-Glass out-of-sample results for a NN and a type-2 FLS designed via a memetic algorithm

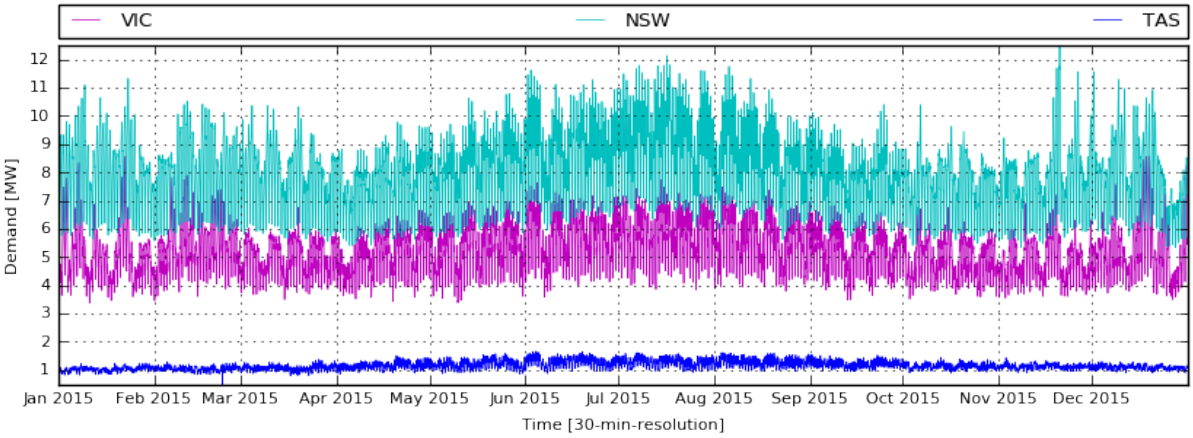

Figure 3.4: Load demands for New South Wales, Victoria & Tasmania regions.

Hence, Table 3.2 results indicate that type-2 FLSs in all cases produced better predictors for the Mackey-Glass time series. Figure 3.3 depicts that the differences between the memetic FLS and the NN occur on the peaks where the NN forecast is not able to adequately model the Mackey-Glass series. Besides, two additional conclusions can then be directly extracted from Table 3.2, the global search performed by the genetic algorithm and the memetic algorithm offered better results due to the fact that BFGS can converge to a local optima. Regarding the system structure, subtractive clustering produced slightly better results than DBSCAN, probably due to the fact that DBSCAN presents problems when separating nearby clusters properly and then the densities vary. Finally, the memetic algorithm managed to obtained a more suited number of rules than the other clustering approach. Nevertheless, all improvements in MAPE decrease as results get closer to optimal solution.

3.4.2 Short-Term Load Forecasting

The objective here is to obtain a one-day-ahead, 30-minutes-resolution load predictor relying only on historical load data. The data-set used corresponds to the Australian electricity market (New South Wales, Victoria & Tasmania regions) electric demands from 2015. As in [26], these regions were selected because their load patterns and scale differ in each month, making testing less subjective. Also the different regions vary in size, being possible to assess how the forecasts evolve according to the market volume (Figure 3.4) and non-linearity (Figures 3.5, 3.6, 3.7). Only historical load data was considered here for the model feature selection, particularly load lags from the previous week.

Although, the consideration of seasonalities, special days or meteorological time series can substantially improve the forecasting accuracy, this testing decision was made in order to show the model's ability to generalise in more challenging conditions. Thus, the last quarter was removed from the training sample to perform the necessary out-of-sample testing. Using only the last quarter for testing as the number of special days and load patterns make the forecasting problem more challenging. That is to say, daily forecasts for the days of the last quarter of 2015 with a 30-minutes-resolution are performed. Besides, as in practice, these will follow a non-recursive procedure to avoid error accumulation, forecasts are produced for $t + 48$ which trivially represents the most challenging horizon. The average of the MAPE over 10 trials is again used here as performance metric.

Several models have been developed using different feature selection mechanisms to compare against the proposed information interaction mechanism (Algorithm 1) and memetic algorithm (Algorithm 2); partial auto-correlation, LASSO, elastic nets, ridge regressors, random forests, all these following a wrapper cross-validation procedure; and MRMR as filter method. To determine the structure, subtractive clustering and DBSCAN, and finally to establish the system parameters again using silhouette; a genetic algorithm and BFGS, which represent a global and local heuristic, respectively. Also, for comparative purposes NNs [23] and SVRs [28] using the mentioned feature selection methods were developed. All these testing procedures are required to assess if indeed there are accuracy improvements by using the proposed feature selection procedure (Algorithm 1) and the proposed variable-length memetic algorithm (Algorithm 2).

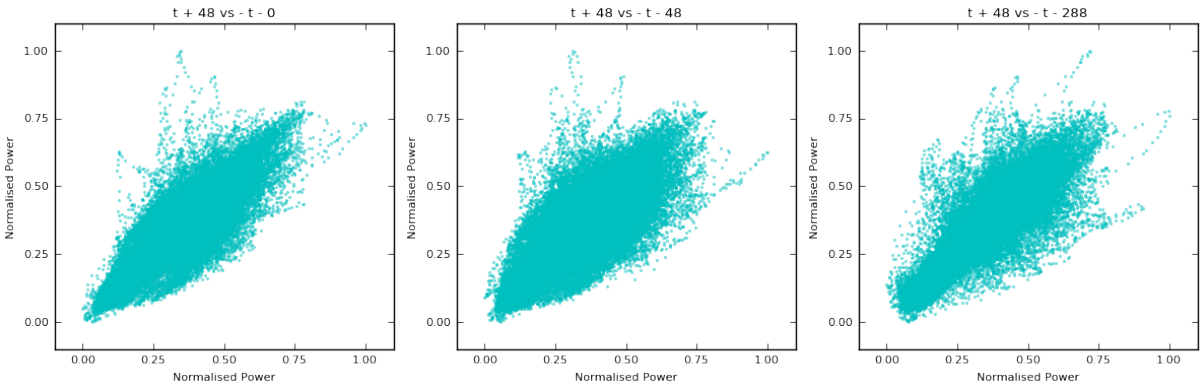

Figure 3.5: New South Wales region output versus input candidates scattergrams.

Datasets are managed by means of pandas library functions and dataframes [117]. A pandas dataframe is a 2-dimensional labeled data structure with columns of not necessarily from the same type. Hence, data is imported into this object which one can easily relate to a tabular object which allows for queries, function mapping. This library also allows to produce figures and compute moving window statistics. Note that in this application due to the volume of data is not necessary to limit the dataframe size and iterate over the dataset.

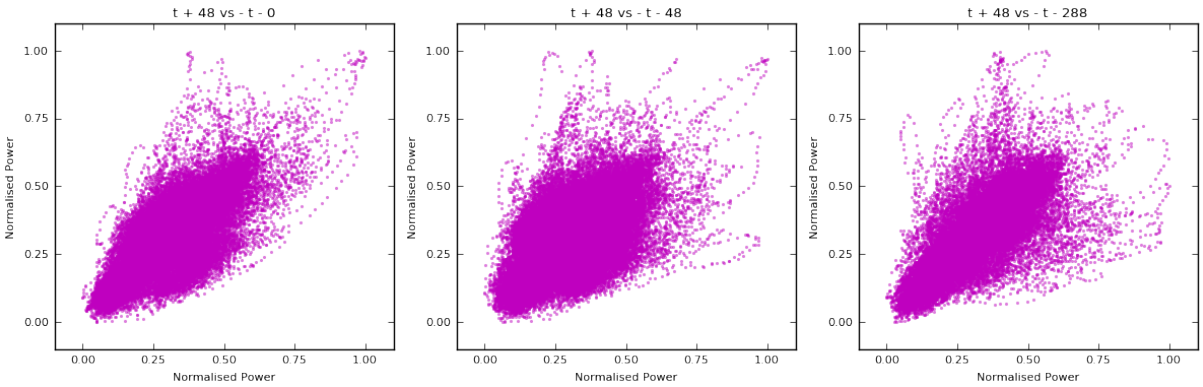

Figure 3.6: Victoria region output versus input candidates scattergrams.

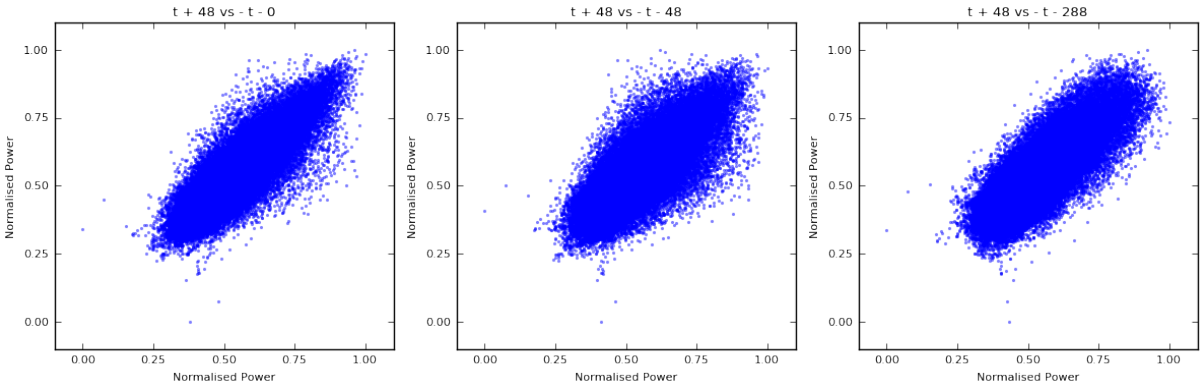

Figure 3.7: Tasmania region output versus input candidates scattergrams.

Figures 3.8 - 3.10 summarise the series out-of-sample forecasting results. Both algorithms, when combined, depict MAPE reductions ranging from 0.07 % to 2.63 %, 0.03 % to 3.19 0.13 % to 0.81 % in the New South Wales, Victoria and Tasmania datasets respectively. Besides, when not combined both are clustered towards the best performance side. When assessing the proposed algorithms individually, accuracy improvements are generally constrained. This is due to the learning dependencies between the system inputs and its structure. Also when improvements are minimal it is due the proposed combination is matching the best achievable performance. Nevertheless, the rest of combinations were not as consistent in performance and represent the impracticalities of a trial-and-error scheme without any guarantee of success. With all the disadvantages that implies in terms of computational costs. Differences between datasets can be explained in Figures 3.5 - 3.7 where regardless the magnitude of the demand the linearity of the sample determines any ability to produce better forecasts and consequently smaller MAPEs are gathered. This also explains why the partial autocorrelation method for feature selection worked better in the Tasmania sample than in the other two. Regarding the

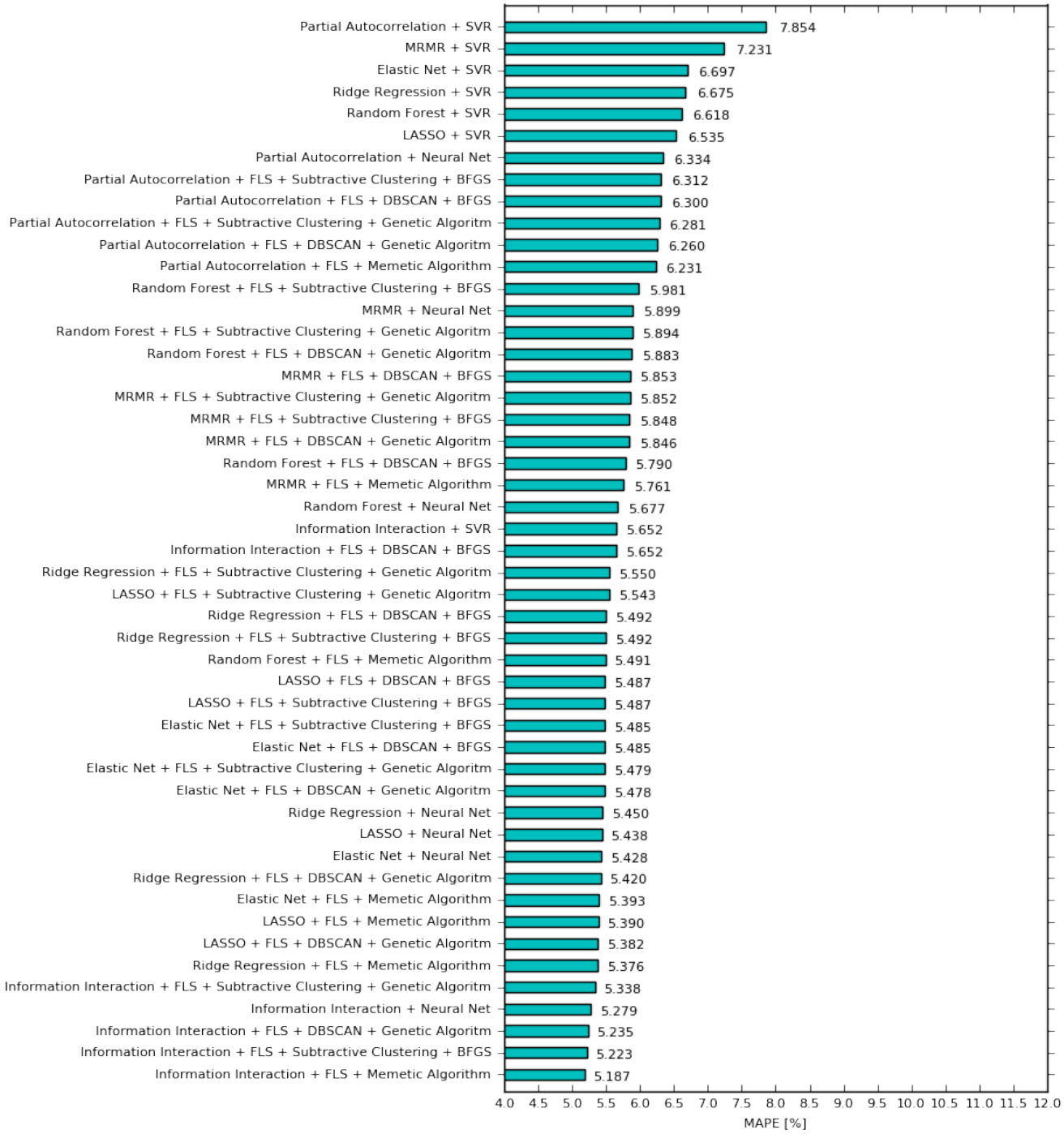

Figure 3.8: Out-of-sample MAPE in the New South Wales dataset

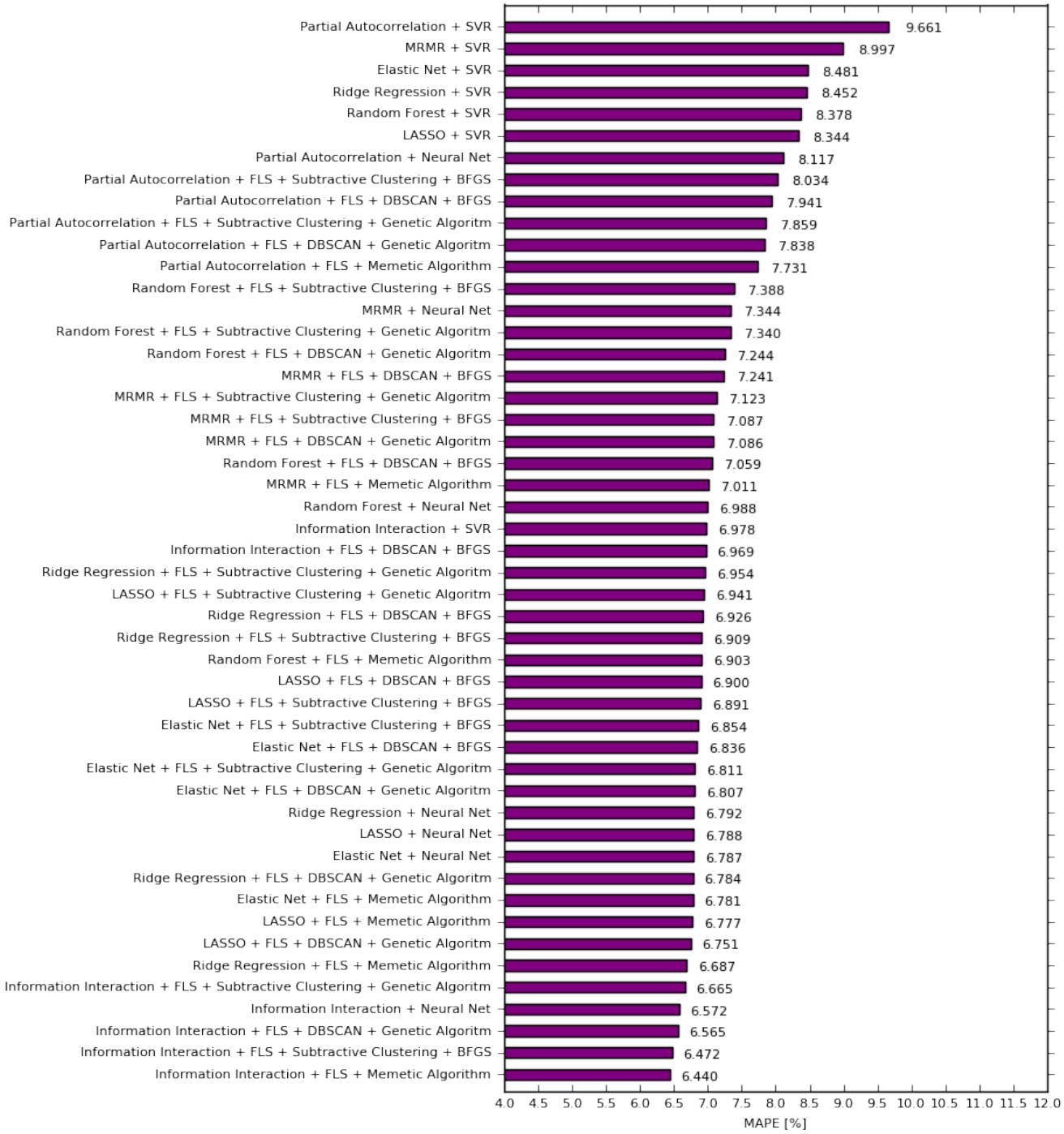

Figure 3.9: Out-of-sample MAPE in the Victoria dataset

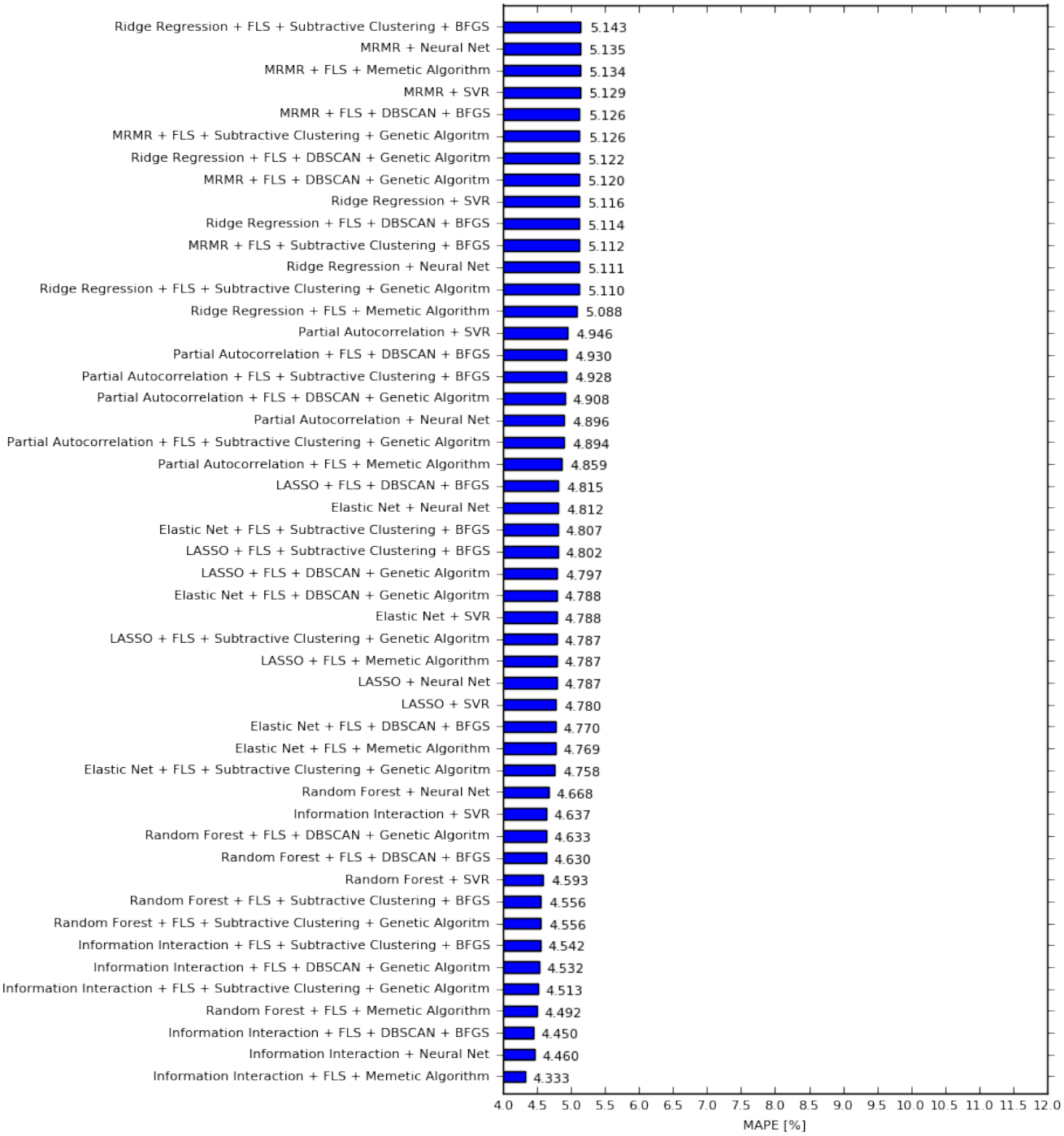

Figure 3.10: Out-of-sample MAPE in the Tasmania dataset

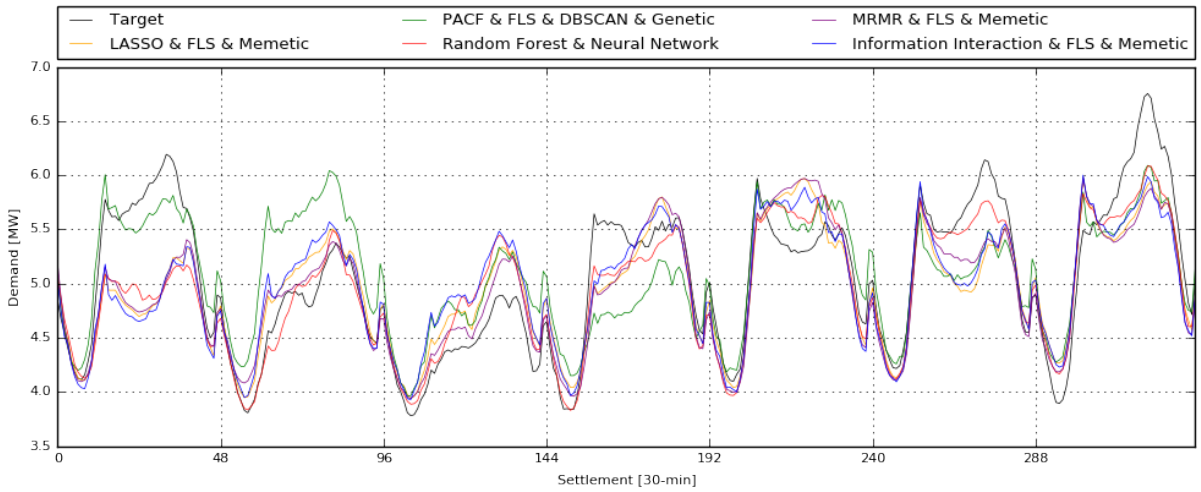

Figure 3.11: One-step-ahead ($t + 48$) forecasts of the different combinations of feature selectors and AI approaches in the Victoria region from 09/10/2015 to 15/10/2015.

performance of other feature selectors, the performance of random forests should be highlighted as they provide the best alternative to the information interaction one. Regarding the latter, the results also illustrate the validity of the introduced information interation feature selector when working with NNs as this combination is also consistently among the best solutions. Nevertheless, it produced inferior performance than the combination information interation feature selector and the memetic FLS approach. Also, SVRs produce worse results than the FLSs and NNs when the the non-linearity in the dataset was higher (i.e. Victoria and New South Wales). In this case, regarding the FLS structure, only memetic algorithm produced consistently better results and differences between subtractive clustering and DBSCAN were not consistent. Finally, as in the Mackey-Glass case, the global search performed by the genetic algorithm and the memetic algorithm offered better results. Neverthess, in this case again differences decrease as results get closer to optimal solution.

3.5 Summary

This chapter has introduced a novel method to automatically design type-2 FLSs for STLF applications. The method is able to identify its constituent features, and concurrently learn its parameters and structure without any kind of initial assumption. This is achieved by means of two novel techniques within the STLF and power systems data-driven domains; an information interaction feature selection procedure (Algorithm 1) and a variable-length memetic learning scheme (Algorithm 2). Moreover, when they are combined as shown in Section 3.4, they can provide MAPE

improvements which consistently provide the best achievable type-2 fuzzy model and accuracy improvement when compared to other combinations of feature selectors, AI-models and a fuzzy learning approaches. As a result of these improvements, gains in power system operational efficiency and market competitiveness can be achieved. But mainly, there is potential to enhance and automate other power systems data-driven applications.

4 Voltage Control in Distribution Networks

Voltage regulation in power systems at distribution level is becoming a challenging problem in terms of control due to new types of demand and intermittent generation. Whilst the goal is to maintain the voltage within the security standards in those operating conditions, an efficient management of the system is as well required. This chapter aims to introduce the use of type-2 FLSs in power systems operation & control, as a suitable scheme for coordinated voltage regulation. Also, its implementation within a hierarchical agent architecture is also investigated. This allows for different levels of coordination between high and low level goals, handling the system's complexity.

4.1 Introduction

Being a key aspect is to obtain timely solutions as closest as possible to the optimal one in order to operate the distribution network as is practicable, achievable and beneficial as possible. The use of traditional mathematical approaches such as robust optimisation become unpractical as scale increases. Hence, following a similar approach to [59], hierarchical type-2 FLSs which incorporate technical, commercial and social perceptions/contexts into the decision-making process can enable performance improvements when compared to other heuristic approaches common in the literature.

This chapter makes a significant contribution as it introduces the use of type-2 FLSs in power systems operation, as a more suitable scheme to cope with significant levels of distributed generation. It appears that little or no other work in the literature has investigated type-2 within this purpose. Furthermore, its implementation within a hierarchical architecture allows for different levels of coordination, handling the system's complexity.

The rest of the chapter is organised as follows: Section 4.2 outlines the chapter contribution, the proposed hierarchical FLSs for voltage control in distribution

networks. In Section 4.3, this is tested and evaluated against other common approaches in literature on three different distribution network models with load and distributed generation profiles. Finally, conclusions and future work are drawn in Section 4.4.

4.2 Hierarchical Type-2 Fuzzy Logic Systems

Type-2 FLSs are especially appropriate to deal with imprecision as they are essentially universal approximators suited to model non-linear continuous processes. All these imply that in terms of power systems, suitable scenarios or devices are the ones where processes or tasks are continuous rather that discrete. This is mainly by the fact that FLSs are inherently continuous and discrete problems might imply a information loss in the response discretisation. Besides, type-2 FLSs rather than type-1 are particularly appropriate for problems hard to model because different non-stationary statistical attributes such us inaccuracies in the voltage sensitivity or the noise measurement cannot be expressed ahead of time mathematically for all conditions. One could easily point out, that given the non-discrete nature of many controllers and their non-linear relation with the network's sensitivity factors, any kind of pre-calculation will lead to inaccuracies. Thus, in those situations where a fast response is needed, computation time can be saved by means of type-2 FLSs. Therefore, their application will probably stand out more clearly in meshed networks rather than radial ones as interactions are less obvious.
Furthermore, in terms of behaviour coordination, type-2 FLSs due to their non-crisp characterisation a smoother response will be obtained due to their less responsive nature when compared to type-1 schemes. Hence, in arbitration situations when conflicting situations arise, this could reduce the number of negotiations. A example of this, could be the efficiency coordination among two different control actions.
However, type-2 FLSs as any other rule-based FLS struggle to handle a vast number of variables (i.e. curse of dimensionality). As the number of rules grows exponentially as it does the number of inputs. This strongly affects its scalability and therefore its application.
Inspired by [59], where a solution is provided to a similar problem in a mobile robots scenario, a hierarchical control problem decomposition is proposed to change the behaviour of rule growth (i.e. it will increase linearly, not exponentially) and provide a

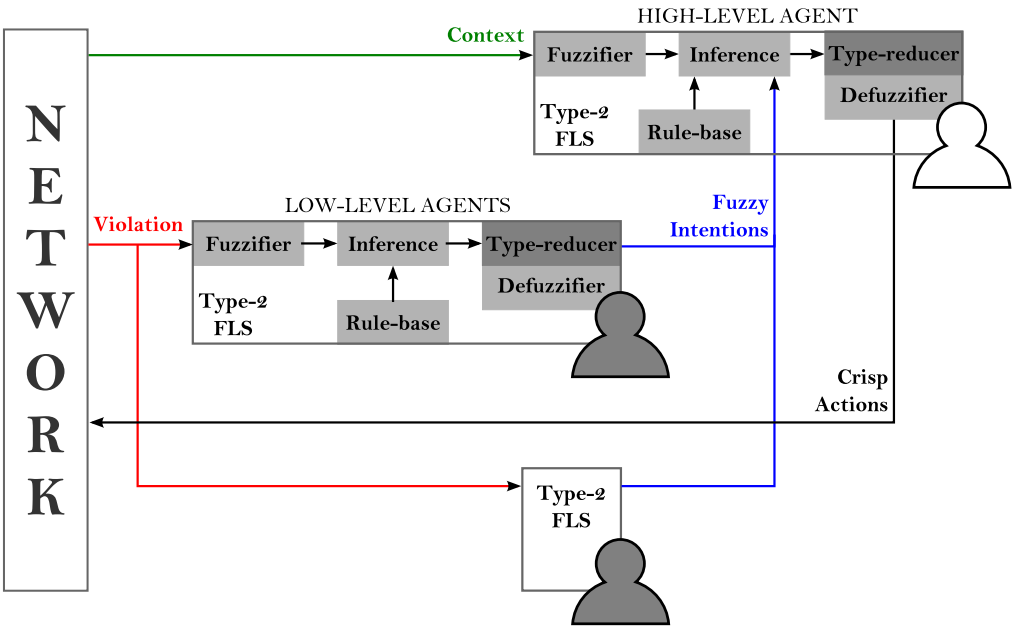

Figure 4.1: Hierarchical Type-2 FLSs.

control scheme where a high-level FLS modulates, according to a specific context, the goals of a collection of low-level FLSs which are working independently. However, low-level FLSs in this case instead of tackling an specific scenario, they are going to be controlling an specific device to achieve a particular goal.

By exploiting this particular hierarchical architecture on power systems in a scalable way, the control task can be tackled in divide-and-conquer fashion and even integrated within a multi-agent system [78, 118], where each agent is a type-2 FLS (Figure 4.1). Thus, by increasing the levels in the structure, introducing new high-level agents above the previous ones, the system has increased capability to operate efficiently in real-time by channelling operations and information flow.

Additionally, according to a range of goals, zones of control or tasks, different levels of coordination can be established (e.g. for a given zone of control which helps limiting the problem complexity, a high-level agent may have the goal to maintain the voltage within the limits, another to perform thermal control. So, by adding another high-level agent above them, coordination between different algorithm goals can be obtained). Furthermore, by increasing the degree of branching of the hierarchical structure, the system can be adapted to the network and by enlarging the number of levels; the system will have greater functionality. Furthermore, this approach allows agents to work in parallel inside the zone, increasing the flexibility to react to each change in the network.

By operating in a similar manner as [55], a solution is produced to fulfil the global goal

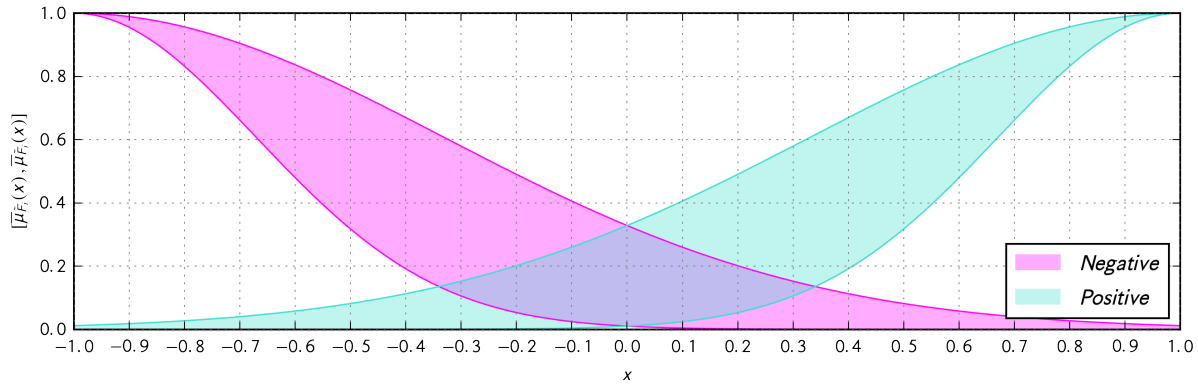

Figure 4.2: Severity, efficiency and availability antecedents.

in a timely fashion. Although here, local goals are also satisfied within a more flexible algorithm that allows for complementary contexts. Type-1 MFs need to be appropriately tuned for each network in order to avoid convergence problems. Also, as in [55] all signals are normalised to $[-1, 1]$. Hence, Algorithm 4 starts by collecting the measurements and fuzzifiying only the most severe voltage violation. The high-level system provides a measure of sensitivity to each low-level system. Then, each individual low-level system reacts to the voltage deviation (i.e. severity) proposing an interval of actions to solve the problem (i.e. type-reduced fuzzy set) depending on their availability and efficiency (i.e. sensitivity) to solve the problem as Table 4.1 illustrates. MFs for high and low antecedents for each of these inputs are depicted in Figure 4.2. The complex three-dimensional interval of intention surfaces, unachievable for their type-1 counterparts are depicted in Figures 4.3 - 4.6, where in blue and are presented the bounds of the type-reduced set as in (2.15). These can be considered as soft mappings of Table 4.1 as relates severity, efficiency and availability. Figures 4.3 - 4.4 illustrate symmetrical intentions depending on the sign of the voltage violation and the local sensitivity factors. Also in both cases, curves saturate towards the extremes showing smooth slopes, which directly translate into smoother controller actions. Figures 4.5 - 4.6 depict control bottlenecks shapes that occur when the controller is at its operational end-points and an intention is not possible. And how the options widen as efficiency indicates a different direction. Also, the final action will be within the red and blue section and will depend on the rest of controllers available and their state. Oscillating between the most helpful action to solve the problem given the control budget and the local utility or desire of each controller.

Given this collection of type-reduced sets, the local goals of the low-level systems

(e.g. reduce number of tap operations, satisfy the desired consumers demand or maximise utility of DGs), and a given context (e.g. decision according to severity or economical targets). This context established whether to satisfy or not the local goals. The way the context works not differs much from a selection priority index [119] which maps the requirements with the control budget and decides which action to deploy (i.e. Table 4.2). For example, the budget is the sum of local sensitivities that is used as an approximation as actions can cancel each other or flows within the network may change affecting voltage levels. Thus, all these inputs are mapped into crisp actions as in [59] and applied and validated using a Matpower network model [45]. This process is continuously re-run until the violation is solved according to the network model, there no changes in the solution or until a maximum number of iterations is reached. Oscillations in voltage magnitude, but also interactions between devices may appear during the solution calculation due to the fact that removing one violation could cause another one to occur. Sequential validation can help avoid trailing problems as the system would backtrack. Also, the maximum number of iterations defines a time limit and was set in order to identify scenarios where a feasible solution cannot be obtained due to the limits of controllability.

Efficiency	Availability	Severity	Intention
Negative	Negative	Negative	Negative
Negative	Negative	Positive	Positive
Negative	Positive	Negative	Zero
Negative	Positive	Positive	Zero
Positive	Negative	Negative	Zero
Positive	Negative	Positive	Zero
Positive	Positive	Negative	Positive
Positive	Positive	Positive	Negative

Table 4.1: Low-Level Rule-Base

Budget	Need	Action
Negative	Negative	Intention
Negative	Positive	Intention
Positive	Negative	Desire
Positive	Positive	Intention

Table 4.2: Context Rule-Base

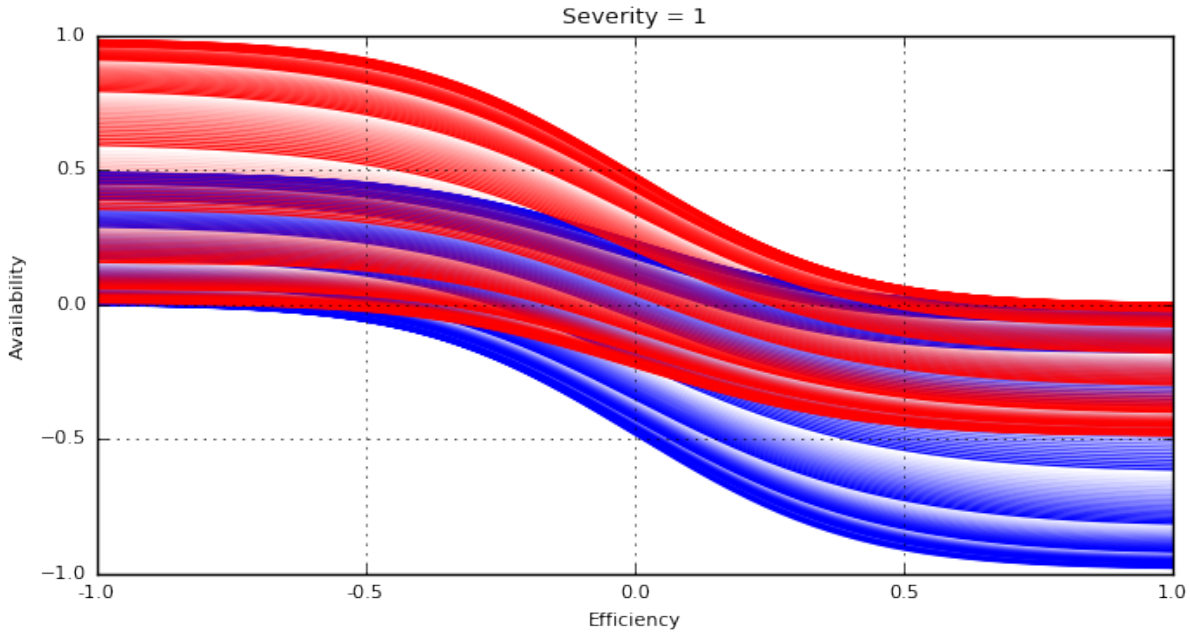

Figure 4.3: Intentions surface for severity = 1

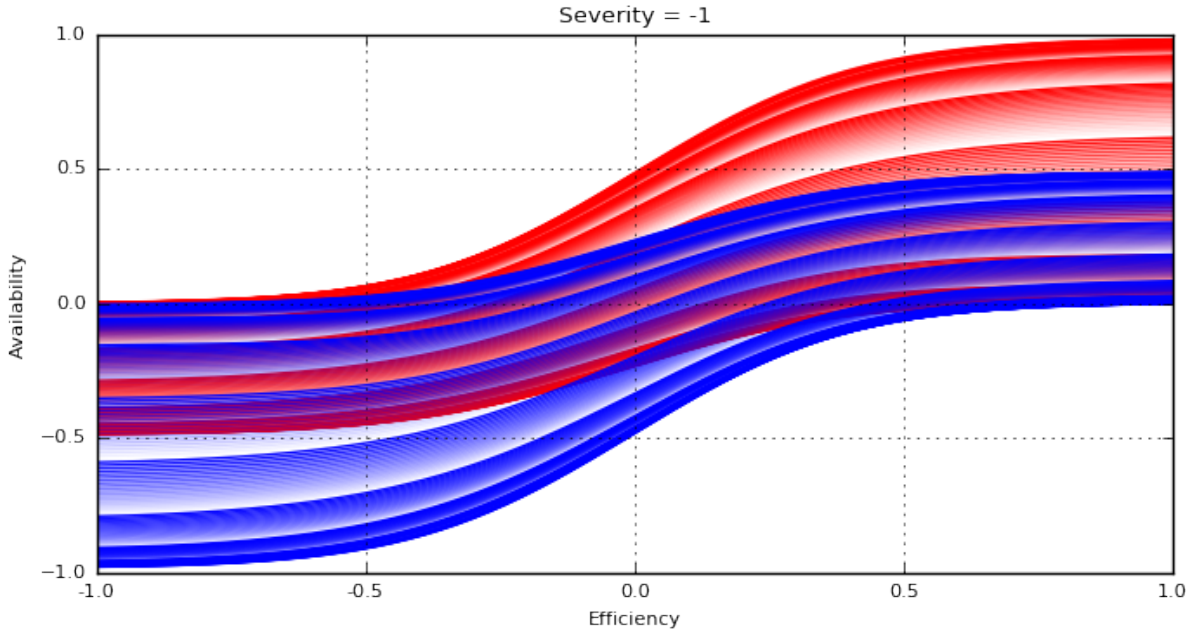

Figure 4.4: Intentions surface for severity = -1

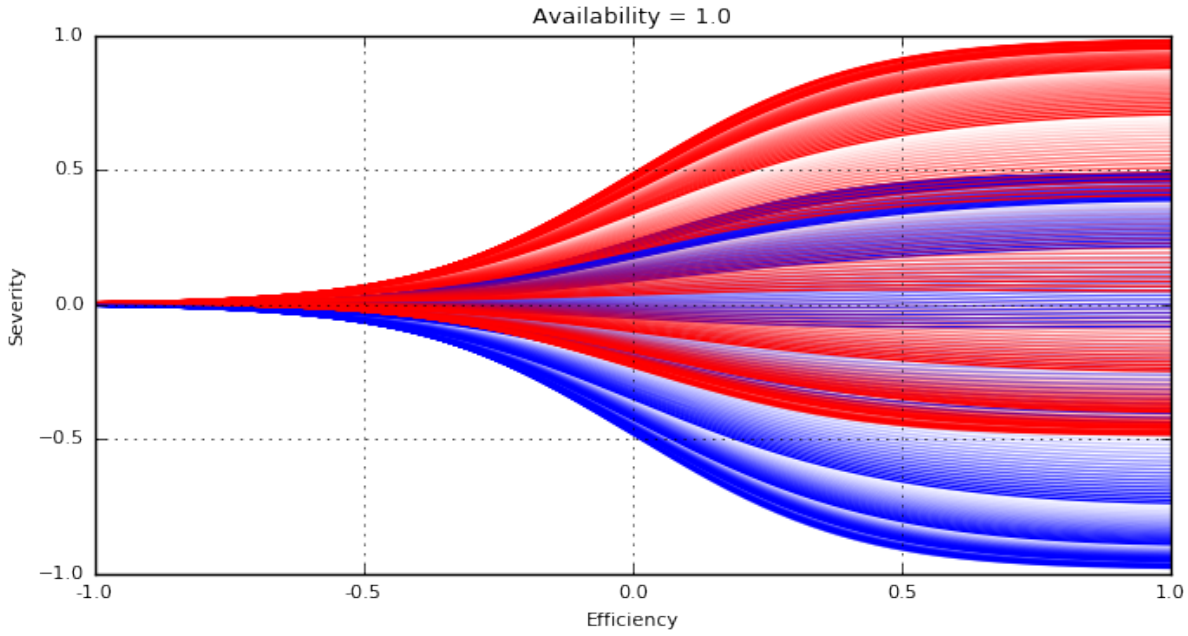

Figure 4.5: Intentions surface for availability = 1

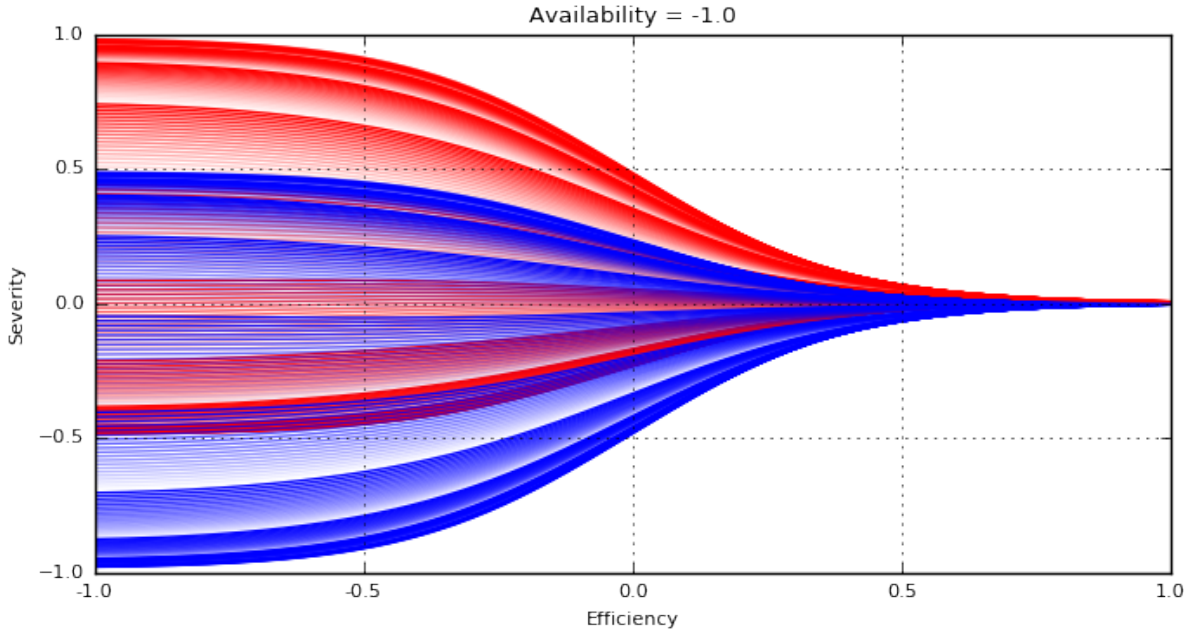

Figure 4.6: Intentions surface for availability = -1

Algorithm 4 Hierarchical Type-2 FLS

1: **Initialisation**
Gather voltage deviations $\{v_1, \ldots, v_n\}$.
2: **while** convergence $\neq$ True **do**
3: **Efficiency retrieval**
Given the worst voltage violation $v^* \in \{v_1, \ldots, v_n\}$ collect devices $\{\frac{\partial v^*}{\partial y_1}, \ldots, \frac{\partial v^*}{\partial y_m}\}$ sensitivity under current conditions.
4: **for** $y_i \in \{y_1, \ldots, y_n\}$ **do**
5: **Intentions computation**
Using the device efficiency $\frac{\partial v^*}{\partial y_i}$, its availability a_i, obtain an interval of intentions $[x_{1,l}, x_{1,r}]$ by means of a type-2 FLS.
6: **Action determination**
Given a set local goals $\{x_1^*, \ldots, x_1^*\}$, a set of intervals of intentions $\{[x_{1,l}, x_{1,r}], \ldots, [x_{m,l}, x_{m,r}]\}$, and a predefined context for arbitration, a set of crisp actions S is determined by means of a type-2 FLS.
7: **return** S

4.3 Experimental Setup, Network Models & Results

The objective here is to ensure voltage deviation at all buses remains within the statutory limits in three different distribution networks; IEEE57, AuRA-NMS 33kV [120] and a reduced version of AuRA-NMS 11kV. The former two are meshed whilst the latter is radial. Different scenarios will be tested in order to evaluated the hierarchical type-2 FLSs performance. In the meshed networks, a comparison only using OLTCs against a type-1 hierarchical FLS, a genetic algorithm, a local controller and greedy algorithm is performed. The first performs a blind-search with a population of 100, a maximum of 100 generations, deterministic tournaments for parent selection and 0.7 and 0.3 as crossover and mutation rates. In the second each OLTC monitors and controls a specific bus according to voltage target and a dead-band around it (i.e, acts as soon as the voltage of this bus abandons this predefined dead-band). The third is a sensitivity-based heuristic. At each iteration selects the most promising action according to the device voltage sensitivity.

Furthermore, in the meshed networks another comparison will be performed also adjusting the real power in DGs. This time only against hierarchical type-1 FLS and a greedy algorithm, as the other are not suited to this mixed-integer non-linear problem. Mixed-integer because of the nature of the different controllers available, that is to say DGs and OLTCs, and non-linear as the squared relation between the power flow into load impedances and applied voltages. Therefore the variables in this voltage control problem are the operational range in which each one of the DGs and OLTCs is, the

most severe voltage violation and the local sensitivity of each one of the DGs and OLTCs towards the most severe voltage violation. Given all these variables, each low-level FLS from this hierarchical scheme will produce an interval of intentions each iteration. These intentions depict the offerings to solve the problem of each FLS, and will be modulated and defuzzified into crisp outputs (e.g. DG output).
In both meshed network cases, the voltage limits are set to ±6% and it is assumed that all the network transformers are OLTCs with 0.01 step size and range ±20. As performance metrics, the number of operations, the largest deviation and curtailment in the cases where DGs are used. Data is also scaled by three in the latter to force coordinated solutions. Finally, only 30 tap moves should occur per OLTC per time period, as more would be unachievable under real conditions. With regards to the radial network, different assumptions are made to increase the case difficulty and to obtain a more challenging scenario in a radial network. The loads and the DGs were considered as totally random, varying uniformly in at each iteration of the 1000 generated. Three type of control devices are here considered, OLTCs, real power control in DGs and demand-side control. Loads at each feeder have different scaling factors to produce unbalanced situations, and have a random ratio to provide demand-side response (i.e. three random values will determine the controllable interval and the desired demand within it). Therefore, the numbers of controllable loads change in every iteration as well as their control conditions, no forecast is considered. Thus, the control objective or global goal is to efficiently maintain the voltage deviation within the limits as well as, reduce number of tap operations, satisfy the desired consumers demand and maximise the utility of DGs. Finally, the voltage limits were set to ±2%. Further details on the specifics of each case are described in following subsections.

4.3.1 IEEE57

This meshed test network has 57 buses, 7 DGs, 15 OLTCs and 42 loads. A few modifications have been made. 4 additional windfarms with capacities 7.5, 9.5, 12 and 14 MVA are connected to buses 33, 57, 30 and 32 respectively using the same scaled half-hourly real windfarm export profile. A schematic of the unmodified network is given in Figure 4.7. Load profile data, also with 30-minutes-resolution have been used [121]. As result, Figure 4.8 depicts the voltage evolution of all 57 buses when no

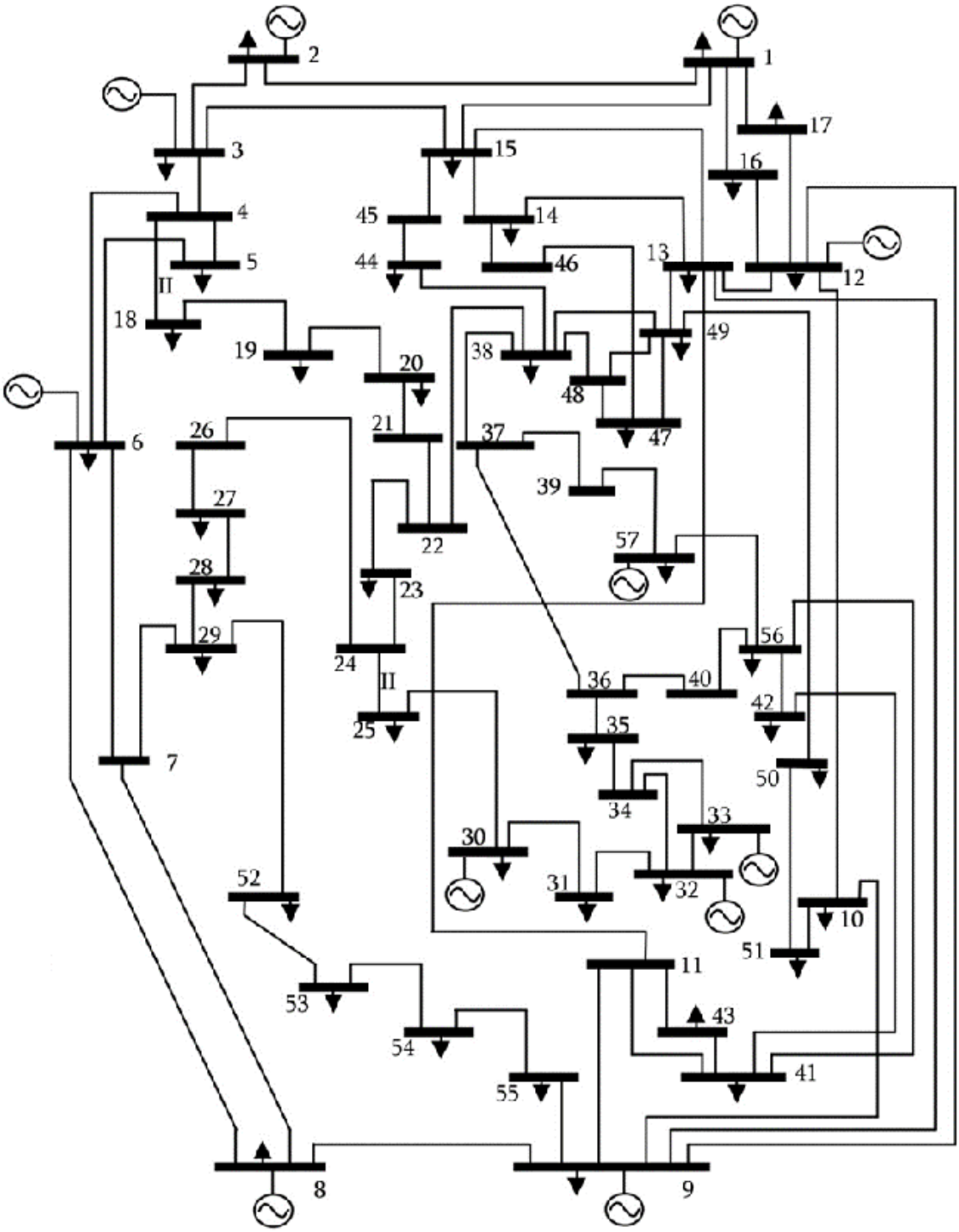

Figure 4.7: IEEE57 Network.

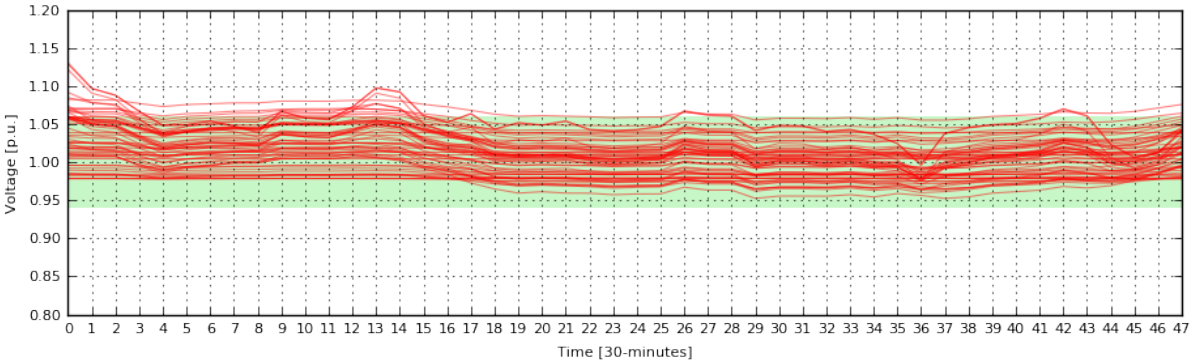

Figure 4.8: IEEE57 - Voltage evolution with no actions performed.

action is perform, just to illustrate that the voltage distribution across all nodes over time is nearly as wide as the operational deadband.

Method	No. Operations	Max. Deviation [p.u.]
Local Control	1062	0.2839
Genetic Algorithm	3623	0.0544
Greedy Algorithm	29	0.0664
Hierarchical Type-1 FLS	34	0.0641
Hierarchical Type-2 FLS	**25**	**0.0596**

Table 4.3: IEEE57 results - OLTCs only.

Method	No. Operations	Max. Deviation [p.u.]	Curtailed [MW]
Greedy Algorithm	38	0.0597	949.517
Hierarchical Type-1 FLS	52	0.0597	980.174
Hierarchical Type-2 FLS	**36**	**0.0596**	**799.801**

Table 4.4: IEEE57 results - OLTCs & DGs.

Tables 4.3 and 4.4 summarise the IEEE57 network results. Type-2 FLSs depict a smoother behaviour with arises from Figures 4.3 - 4.6 with lower number of tap operations and fewer curtailment of the DGs. All fulfilling the goal to keep the voltage within the limits except for the local controllers. Operating with fixed targets, they continuously trailed due to their interactions. That is to say, OLTCs overreacted and cancel each other because of the meshed nature of the network and the not obvious sign of the local sensitivity factors. The genetic algorithm had convergence issues and did not provide timely solutions as it required an unachivable number of operations. Type-1 FLSs obtained similar performance as the greedy approach although with a greater number of operations as the MFs were not optimal. This occurred in both scenarios, with and without DGs. Finally, the hierarchical Type-2 FLS not only managed to achieve the lowest voltage violation but as well with the lowest number of

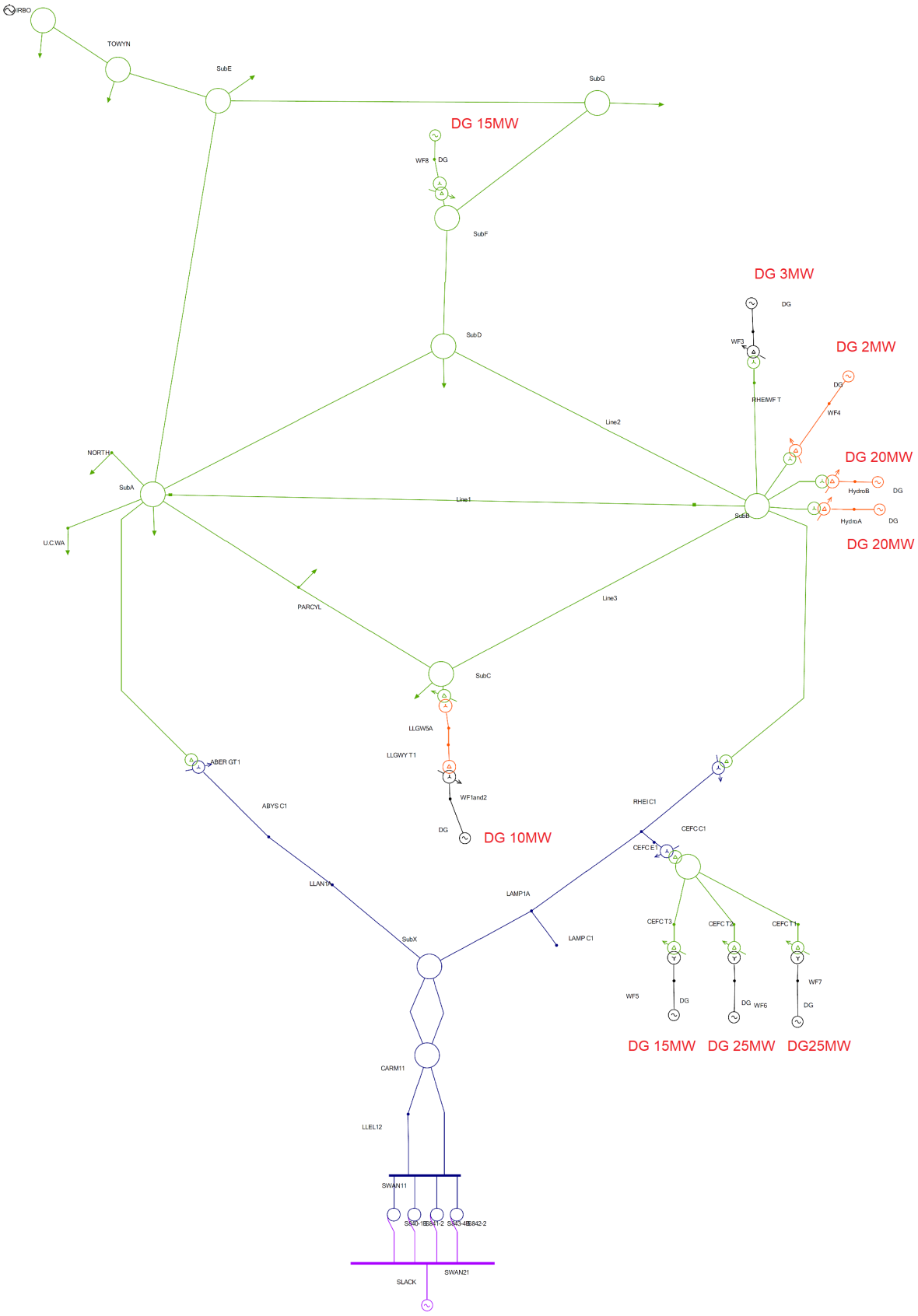

Figure 4.9: AuRA-NMS 33kV Network.

operations and curtailing an inferior amount of generation.

4.3.2 AuRA-NMS 33kV

This meshed test network [120] has 40 buses, 10 loads and 9 DGs, 2 hydro and 9 windfarms. Hydro plants with capacities of 20 MVA each one, and windfarms with capacities, 2, 3, 10, 15, 15, 20, 20 and 25 MVA respectively. A schematic is given in Figure 4.9. Again time-series data has 30-minutes and except for the hydro are profiles are the same as in the IEEE57 case. Also, the network has 19 OLTCs with the same characteristics. Figure 4.10 depicts the voltage evolution of all 40 buses without any action being perform to again illustrate that in this case the network is heavily loaded.

Method	No. Operations	Max. Deviation [p.u.]
Local Control	7058	0.2839
Genetic Algorithm	4682	0.0599
Greedy Algorithm	32	0.6000
Hierarchical Type-1 FLS	43	0.0965
Hierarchical Type-2 FLS	**31**	**0.0596**

Table 4.5: AuRA-NMS 33kV results - OLTCs only.

Method	No. Operations	Max. Deviation [p.u.]	Curtailed [MW]
Greedy Algorithm	46	**0.0596**	820.449
Hierarchical Type-1 FLS	102	**0.0596**	788.912
Hierarchical Type-2 FLS	**41**	**0.0596**	**776.146**

Table 4.6: AuRA-NMS 33kV results - OLTCs & DGs.

Tables 4.5 and 4.6 illustrate the AuRA-NMS 33kV network results. Similar conclusions can be obtained. As in the IEEE57 case, the local and genetic approaches can be discarded because of the very same reasons. The local control did not manage to solve the voltage control problem and the genetic algorithm did not do it in a timely manner. Type-2 FLSs produce again an improved solution in terms of operation and the amount of curtailed generation. Furthermore, in the OLTCs only case was the only one the achieve an adequate solution.

4.3.3 AuRA-NMS 11kV

This test network is a simplification of an existing 11kV UK radial network from the AuRA-NMS project [120]. The network has 3 OLTCs are dependant in their operation,

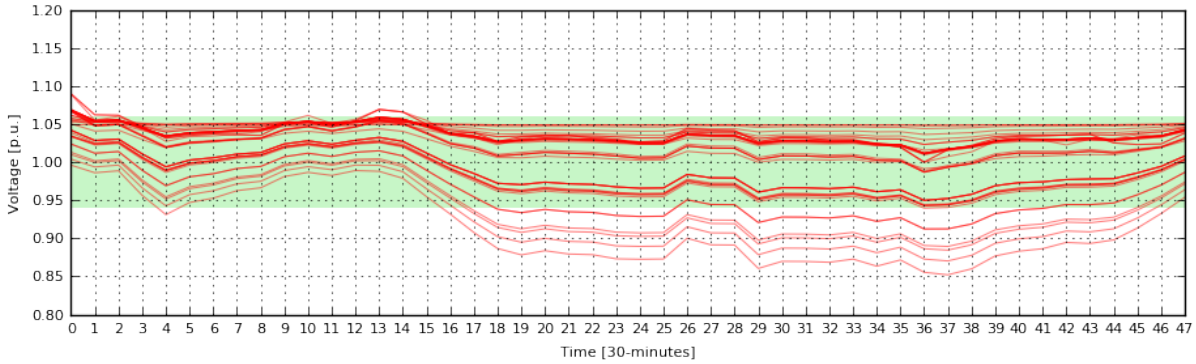

Figure 4.10: AuRA-NMS 33kV - Voltage evolution with no actions performed.

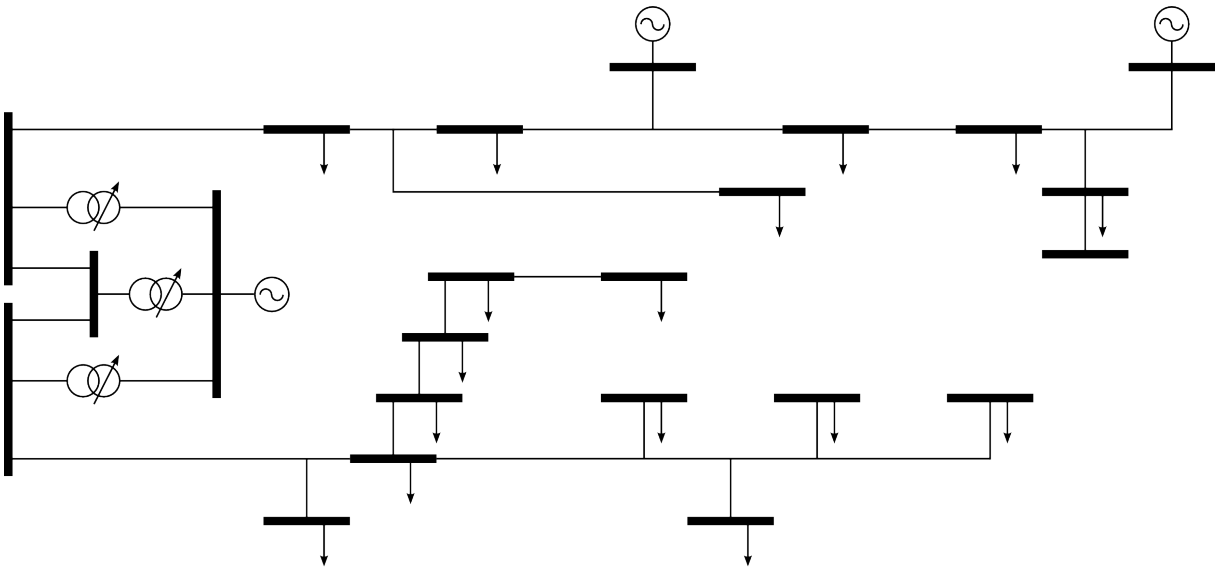

Figure 4.11: AuRA-NMS 11kV Network.

40 buses and is composed of two radial feeders, each with several loads. The top feeder also has two DGs. Both are connected to a grid infeed at 33 kV through three transformers.

In Figure 4.12 simulation results are shown. Note the existing variability on the most severe voltage violation along iterations, neither load nor generation dominant, is mainly explained by the inherent uncertainty in the case generation (i.e. a sequence of random scenarios). Nonetheless, hierarchical type-2 FLSs were capable of maintaining all voltages within ±2% on the 1000 tested iterations using pre-computed sensitivity factors in less than 5 iterations per case. According to the obtained results, the objective of obtaining timely solutions to his coordinated problem was successfully achieved. Hierarchical type-2 FLSs were able to adapt to controllability changes without modifying their initial parameters or rule-base.

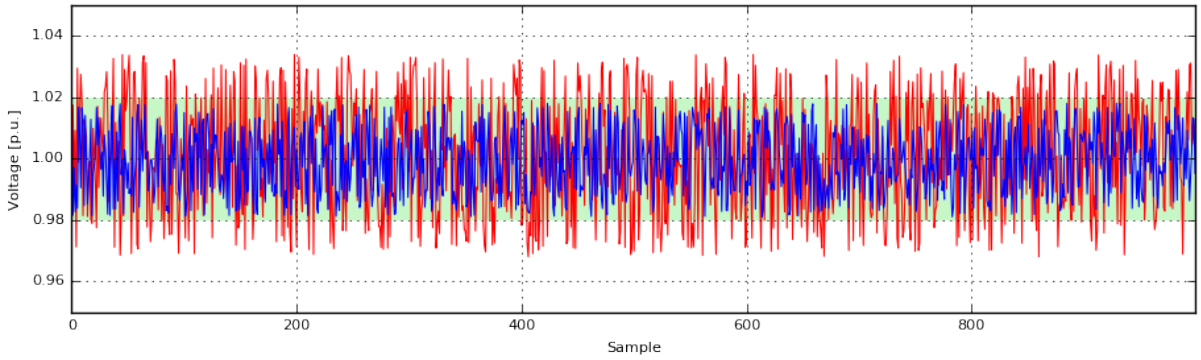

Figure 4.12: AuRA-NMS 11kV - Worst voltage deviation evolution with no actions performed and with the hierarchical type-2 FLS.

4.4 Summary

This chapter has introduced a novel method to flexible address voltage control in distribution networks. Thus, according to the obtained results, coordination between devices, global and local objectives was successfully achieved (e.g. maintain voltage within the limits whilst reducing the number of tap operations, satisfying and maximising the utility of DGs). Type-2 FLSs were able to consistently over-performed their type-1 counterparts and other alternatives such a greedy sensitivity-based search. As a result of these improvements, operational life of OLTCs can be extended and/or DG penetration increased.

5 Conclusions

Type-2 FLSs have many substantial advantages when compared to their type-1 counterparts and provide a smoother control due to the smother shape of the control surface [71]. The modulated response is less abrupt when compared to type-1 FLS because of the use of interval firings. Hence, when compared to type-1 FLS a more adaptive system can be obtained by allowing more complex relationships as Figures 4.3 - 4.6 depicted. Type-1 FLSs have to be considered as merely reduced versions of type-2 FLSs where the upper and lower MFs are the same. Another advantages worth to mention is that the input range can be covered with a fewer number of sets obtaining similar input-output mappings, this means that with fewer degrees of freedom similar relationships can be achieved. Furthermore, the offer an increase of the system robustness and handle better disturbances around the controlling point. Therefore, in the power system case, like in many other real world applications, type-2 FLSs offer capable solutions to deal with non-linear mappings where imprecision is inherent.

Besides, how these are designed is essential. Being able to identify its constituent features, and concurrently learn its parameters and structure without any kind of initial assumption, enables performance improvements. This can be achieved by means of two novel techniques introduced in this thesis that are tailored to the STLF domain; an information interaction feature selection procedure (Algorithm 1) and a variable-length memetic learning scheme (Algorithm 2). Results in Section 3.4 depict these improvements in terms of MAPE that can unveil gains in power system operational efficiency and market competitiveness. These to algorithms not only consistently provide the best achievable type-2 fuzzy model and MAPE when compared to other combinations of feature selectors, AI-models and a fuzzy learning approaches. But also, enable the opportunity to enhance and automate other power systems data-driven applications.

Finally when dealing with voltage control in distribution networks. According to the

results in Section 4.3, coordination between devices maximising local utility is possible by means of a hierarchical implementation of type FLSs. The voltage can be maintained within the operational limits whilst minimising the OLTC operations and DG curtailment. As a result of these improvements, OLTCs operational life can be extended as well as the environmental and economical benefits of DGs can be increased.
The rest of the chapter enumerates this thesis contributions and key findings. Additionally, discusses broader implications and future research directions that could effectively extend this work.

5.1 Contributions

This thesis contributes with a novel framework to automatically design type-2 fuzzy logic systems for modelling applications, and load forecasting in particular. This along with the voltage control for distribution networks implementation bring to light the capabilities of type-2 fuzzy logic systems to directly derive benefits to energy stakeholders in two distinctive areas.

1. An information interaction feature selection procedure is introduced, novel within the power systems domain that as more generic approach overcomes the limitations of other entropy-based selectors. Also, appropriately selecting features allows for improving accuracy and decreases training time.

2. A memetic algorithm to concurrently learn the structure and parameters of a type-2 fuzzy logic systems is introduced, novel within the model identification domain. It consistently provides superior performance unconstraining fuzzy logic systems modelling capabilities.

3. A scalable implementation of type-2 fuzzy logic systems that allows the voltage control task to be tackled in divide-and-conquer fashion and even integrated within a multi-agent systems.

Also of importance, within each of these mechanisms, through the proposed formulations and solution approaches, several smaller research contributions arise from the present work;

1. Load forecasting comparison in distinctive market regions of a combination of feature selectors and artificial intelligence algorithms and fuzzy learning schemes.

2. Use of the BFGS gradient-based parameter optimisation and jacobian definition for the its implementation on type-2 fuzzy logic systems which use Nie-Tan type-reduction.

3. Use of entropy metrics to compute a measure of fuzzy rule similarity in an empirical fashion.

4. Messy coding implementation for type-2 fuzzy logic systems that allows for concurrent optimisation of systems with different numbers of rules.

5. Comparison of type-2 fuzzy logic systems for voltage control against common approaches and its type-1 counterpart in 2 different distribution meshed networks.

5.2 Broader Implications & Future Research

1. There are many other tasks that require precise forecasting techniques (e.g. wholesale prices or wind generation among other) [122, 123]. Type-2 fuzzy logic systems have demonstrated their performance on short-term load forecasting when thesis contributions were used. Then, it could exploited in all these other challenging time-series forecasting problems.

2. Extending the load forecasting automated design methodology to enable probabilistic forecasting via direct learning of prediction intervals. This has many applications especially in wind forecasting to improve realiability [124].

3. Another interesting problem is to explore load forecasting at sub-regional levels and assess how small the sample can before the predictive performance will start to degrade. This will open opportunities to aggregators and suppliers to further optimise their portfolio.

4. Integration of all the contributions introduced here, to enable model predictive control, demand-response scheduling for services provision or even through

learning explore how to avoid the need for using a load flow engine as a validator.

A Source Code

```
  language
1 import pandas                 as    pd
2 import numpy                  as    np
3 from  copy                    import deepcopy
4 from  scipy                   import optimize, spatial
5 from  random                  import choice, randrange, random
6 from  sklearn.utils.random    import sample_without_replacement
7 from  scipy.spatial           import cKDTree
8 from  scipy.special           import digamma
9 from  pypower.runpf           import runpf
10
11 def avgdigamma(points, dvec):
12     # This part finds number of neighbors in some radius in the marginal space returns
           expectation value of <psi(nx)>
13     N = len(points)
14     tree = cKDTree(points)
15     avg = 0.
16     for i in range(N):
17         dist = dvec[i]
18         num_points = len(tree.query_ball_point(points[i],dist-1e-15,p=float('inf')))
19         avg += digamma(num_points) / N
20     return avg
21
22 def mi (X, Y, k = 3, base = 2.718281828459045):
23     # Mutual information 'k'-nearest neigbors estimator for the time-series 'X' and 'Y'
            which are 1D-arrays. Add noise to input time-series to break degeneracy
24     x     = (X + 1e-10 * np.random.rand(len(X))).reshape(len(X), 1)
25     y     = (Y + 1e-10 * np.random.rand(len(Y))).reshape(len(Y), 1)
26     # Define joint space
27     xy    = np.hstack([x,y])
```

```
28      # Define kd-tree for quick nearest-neighbor lookup
29      tree  = cKDTree(xy)
30      # Find k-nearest neighbors distances for every point in 't' using the 'Minkowski'
            distance with 'p = inf'
31      dvec  = [tree.query(point, k + 1, p = float('inf'))[0][k] for point in xy]
32      # Return the average number of neighbors of each point from 'j' according to its
            distance 'd' in the marginal space
33      return (-avgdigamma(x, dvec) - avgdigamma(y, dvec) + digamma(k) + digamma(len(x)))
            / np.log(base)
34
35  def cmi (X, Y, Z, k = 3, base = 2.718281828459045):
36      # Conditional Mutual information 'k'-nearest neigbors estimator for the time-series
             'X', 'Y' and 'Z' which are 1D-arrays. Add noise to input time-series to break
            degeneracy
37      x     = (X + 1e-10 * np.random.rand(len(X))).reshape(len(X), 1)
38      y     = (Y + 1e-10 * np.random.rand(len(Y))).reshape(len(Y), 1)
39      z     = (Z + 1e-10 * np.random.rand(len(Z))).reshape(len(Z), 1)
40      # Define joint spaces
41      xyz   = np.hstack([x,y,z])
42      xz    = np.hstack([x, z])
43      yz    = np.hstack([ y,z])
44      # Define kd-tree for quick nearest-neighbor lookup
45      tree  = cKDTree(xyz)
46      # Find k-nearest neighbors distances for every point in 't' using the 'Minkowski'
            distance with 'p = inf'
47      dvec  = [tree.query(point, k + 1, p = float('inf'))[0][k] for point in xyz]
48      # Return the average number of neighbors of each point from 'j' according to its
            distance 'd' in the marginal space
49      return (-avgdigamma(xz, dvec) - avgdigamma(yz, dvec) + avgdigamma(z, dvec) +
            digamma(k)) / np.log(base)
50
51  def info_interaction (training, reduction=10, beta=1.0, gamma=0.25):
52      # Prepare data
53      x = training[[k for k in range(training.shape[1]) if 'Demand +' not in training.
            columns[k]]].as_matrix()
54      y = training['Demand + 48'].as_matrix()
55      # Get subsample
```

```
56      s  = sample_without_replacement(x.shape[0], x.shape[0] / reduction)
57      # Rank all candidates
58      ranking = np.array(sorted([[mi(x[s,j], y[s]), j] for j in range(x.shape[1])], key=
            lambda j:j[0], reverse=1))
59      # Define initial subset
60      subset = [ranking[0][1]]
61      # Compute information for input-output, input-to-input & input-input given 'output'
             according to 'subset'
62      xy ={k[1]:k[0] for k in [[ mi(x[s,j[1]],y[s]),j[1]] for j in ranking[1:]]}
63      xx ={subset[-1]:{k[1]:k[0] for k in [[ mi(x[s,j[1]],x[s,subset[-1]]),j[1]] for j in
             ranking[1:]]}}
64      xxy={subset[-1]:{k[1]:k[0] for k in [[cmi(x[s,j[1]],x[s,subset[-1]],y[s]),j[1]] for
             j in ranking[1:]]}}
65      # Explore subsets with cardinality lower than 20
66      for i in range (20 - 1):
67          ranking = sorted([p for p in [[xy[k[1]] + sum([- beta * xx [j][k[1]] + gamma *
                xxy[j][k[1]] for j in subset]), k[1]] for k in ranking[1:]]], reverse =
                True)
68          # If empty, then stop
69          if ranking == []: break
70          # Compute Hampel distances for termination criterion
71          d = np.abs([k[0] - np.median([k[0] for k in ranking]) for k in ranking])
72          # If not an outlier, then stop
73          if (d[0] / (1.4826 * np.median(d))) <= 3: break
74          # Otherwise, add & continue
75          subset +=[ranking[0][1]]
76          xx [subset[-1]] ={k[1]: k[0] for k in [[ mi(x[s,j[1]],x[s,subset[-1]]),j[1]]
                for j in ranking[1:]]}
77          xxy[subset[-1]] ={k[1]: k[0] for k in [[cmi(x[s,j[1]],x[s,subset[-1]],y[s]),j
                [1]] for j in ranking[1:]]}
78      # Return
79      return {'features' : training[[k for k in range(training.shape[1]) if 'Demand +'
            not in training.columns[k]]].columns[subset]}
80
81  def generate (no_rules, no_inputs): return [add_rule (no_inputs) for k in xrange(
        no_rules)]
82
```

```
83 def add_rule (no_inputs):
84     return [np.sort(np.random.rand(no_inputs+1, 1))[:,::-1], np.hstack([np.random.rand(
           no_inputs, 1), np.sort(np.random.rand(no_inputs , 2) * 0.33, axis=1)[:,::-1]])]
85
86 def flat (system):
87     return np.hstack([np.hstack([system[k][0].flatten(), system[k][1].flatten()]) for k
            in range(len(system))])
88
89 def hierarchical (system, no_inputs):
90     # Rules division given the no. parameters per rule
91     rule = np.array([system[k:k+(4*no_inputs+1)] for k in range(0, len(system), (4*
           no_inputs+1))])
92     # Consequents and Antecedents division
93     return [[rule[k][:no_inputs+1 ].reshape(no_inputs+1,1), rule[k][ no_inputs+1:].
           reshape(no_inputs ,3)] for k in range(len(rule))]
94
95 def prediction (system, x, y, no_inputs):
96     # Adjust structure
97     system    = hierarchical (system , no_inputs)
98     no_rules  = len(system)
99     # Get errors
100    firing_up = np.array([np.exp(-0.5 * (x.T - system[k][1][:,0]) ** 2.0 / (system[k
           ][1][:,1]) ** 2.0).prod(axis=1).T for k in xrange(no_rules)])
101    firing_lo = np.array([np.exp(-0.5 * (x.T - system[k][1][:,0]) ** 2.0 / (system[k
           ][1][:,2]) ** 2.0).prod(axis=1).T for k in xrange(no_rules)])
102    output    = np.array([system[k][0][:,0].dot(np.vstack([np.ones([1, x.shape[1]]), x
           ])) for k in xrange(no_rules)])
103    return np.nan_to_num(((firing_up * output).sum(axis=0) + (firing_lo * output).sum(
           axis=0)) / (firing_up + firing_lo).sum(axis=0))
104
105 def jacobian (system, x, y, no_inputs):
106    # Adjust structure
107    jacobian  = hierarchical (np.zeros_like(system), no_inputs)
108    system    = hierarchical (           system , no_inputs)
109    no_rules  = len(system)
110    # Get errors
```

```
111     firing_up = np.array([np.exp(-0.5 * (x.T - system[k][1][:,0]) ** 2.0 / (system[k
            ][1][:,1]) ** 2.0).prod(axis=1).T for k in xrange(no_rules)])
112     firing_lo = np.array([np.exp(-0.5 * (x.T - system[k][1][:,0]) ** 2.0 / (system[k
            ][1][:,2]) ** 2.0).prod(axis=1).T for k in xrange(no_rules)])
113     output    = np.array([system[k][0][:,0].dot(np.vstack([np.ones([1, x.shape[1]]), x
            ])) for k in xrange(no_rules)])
114     prediction = np.nan_to_num(((firing_up * output).sum(axis=0) + (firing_lo * output)
            .sum(axis=0)) / (firing_up + firing_lo).sum(axis=0))
115     error     = (prediction - y)
116     # Compute derivatives for each parameter of each rule
117     rule      = (output - prediction) / (firing_up + firing_lo).sum(axis=0)
118     l_sample  = np.vstack([np.ones([1, x.shape[1]]), x])
119     options   = [np.array([l for l in range(0, x.shape[0]) if l != n]) for n in range
            (0, x.shape[0])]
120     for k in range(len(system)):
121         # Consequents
122         jacobian[k][0]     = 2.0 * (error * (l_sample * (firing_up[k] + firing_lo[k]))
                / (firing_up + firing_lo).sum(axis=0)).mean(axis=1)
123         # Antecedents
124         if len(options)   == 1:
125             rest_up        = 1.0
126             rest_lo        = 1.0
127         else:
128             rest_up = np.array([(np.exp(-0.5*((x[o,:].T - system[k][1][o,0]) ** 2.0 / (
                    system[k][1][o,1]) ** 2.0)).T).prod(axis=0) for o in options])
129             rest_lo = np.array([(np.exp(-0.5*((x[o,:].T - system[k][1][o,0]) ** 2.0 / (
                    system[k][1][o,2]) ** 2.0)).T).prod(axis=0) for o in options])
130         gaussian_up = np.exp(-0.5 * ((x.T - system[k][1][:, 0]) ** 2.0 / (system[k
                ][1][:, 1]) ** 2.0)).T
131         gaussian_lo = np.exp(-0.5 * ((x.T - system[k][1][:, 0]) ** 2.0 / (system[k
                ][1][:, 2]) ** 2.0)).T
132         exp_mean_up = ((x.T - system[k][1][:, 0]) / (system[k][1][:, 1]) ** 2.0).T
133         exp_mean_lo = ((x.T - system[k][1][:, 0]) / (system[k][1][:, 2]) ** 2.0).T
134         exp_sigm_up = ((x.T - system[k][1][:, 0]) ** 2.0 / (system[k][1][:, 1]) ** 3.0)
                .T
135         exp_sigm_lo = ((x.T - system[k][1][:, 0]) ** 2.0 / (system[k][1][:, 2]) ** 3.0)
                .T
```

```
136         jacobian[k][1][:,0] = 2.0*(error*rule[k]*(rest_up*gaussian_up*exp_mean_up +
                rest_lo*gaussian_lo*exp_mean_lo)).mean(axis=1)
137         jacobian[k][1][:,1] = 2.0*(error*rule[k]*(rest_up*gaussian_up*exp_sigm_up)).
                mean(axis=1)
138         jacobian[k][1][:,2] = 2.0*(error*rule[k]*(rest_lo*gaussian_lo*exp_sigm_lo)).
                mean(axis=1)
139     # Return jacobian in flat-form
140     return flat(jacobian)
141
142 def mse (system, x, y, no_inputs):
143     # Return Mean Squared Error
144     return ((prediction (system, x, y, no_inputs) - y) ** 2.0).mean()
145
146 def fls (training, testing, p, q, no_rules, features, method:
147     # Define model
148     no_inputs = len(features)
149     system   = flat(generate (no_rules, no_inputs))
150     x        = (training[features].T).as_matrix()
151     y        = (training['output'] ).as_matrix()
152     if   method == 'BFGS':
153         system = minimize(mse, system, args = (x, y, no_inputs), method = 'BFGS', jac =
                jacobian, options = {'maxiter': 200})['x']
154     else:
155         system = memetic (x, y, no_inputs)
156     # Get predictions
157     y   = testing['t + 6'].as_matrix()
158     y_  = prediction(system, (testing[features].copy().T).as_matrix(),(testing['output
            '].copy() ).as_matrix(), no_inputs)
159     y   = (y  * (p - q)) + q
160     y_  = (y_ * (p - q)) + q
161
162     # Return
163     return {'prediction': y_,
164             'RMSE'     : (((y - y_) ** 2.0).mean()) ** 0.5,
165             'MAPE'     : np.round(100*(np.absolute(np.divide((y - y_), y))).mean(), 5),
166             'FLS'      : system}
167
```

```
168 def memetic (x, y, no_inputs, iterations=50, size=20):
169     # Initial population
170     population = [generate(2, no_inputs) for k in range(size)]
171     # Ranking & Elite
172     ranking   = sorted([[fitness(k, x, y, no_inputs), k] for k in population], key=
            lambda j: j[0], reverse = False)
173     elite     = ranking[0]
174     # Loop
175     for i in range(iterations):
176         # Only if it's not the very last one
177         if i != iterations - 1:
178             # Check elite
179             elite     = deepcopy(ranking[0]) if ranking [0][0] < elite[0] else deepcopy(
                    elite)
180             # Distribute in pairs
181             selection = deepcopy(ranking) ; np.random.shuffle(selection)
182             selection = [[selection[i], selection[i+1]] for i in range(0, size, 2)]
183             # Next generation
184             population = [evolve (i, x, y, no_inputs) for i in selection]
185             # Rank
186             ranking   = [k   for k in population]
187             ranking   = [k[0] for k in ranking] + [k[1] for k in ranking]
188             ranking   = sorted(ranking, key=lambda j: j[0], reverse = False)[:size]
189     # Return
190     return flat(elite[1])
191
192 def grouper(sequence):
193     result = [] # will hold (members, group) tuples
194     for item in sequence:
195         for members, group in result:
196             if members.intersection(item): # overlap
197                 members.update(item)
198                 group.append(item)
199                 break
200         else: # no group found, add new
201             result.append((set(item), [item]))
202     # Unify
```

```
203     lst = [group for members, group in result]
204     for i in range(len(lst)):
205         if len(lst[i]) == 1: lst[i] = lst[i][0]
206         else:
207             tmp = []
208             for j in lst[i]: tmp += j
209             lst[i] = list(set(tmp))
210     return lst
211
212 def fitness (system, x, y, no_inputs, alpha=5.0):
213     # Return AIC
214     return np.log(mse (flat(system), x, y, no_inputs)) + alpha * (len(system) / np.
            float(x.shape[1]))
215
216
217 def switch_point (d,x):
218     #Returns the switch point of a list 'd' given 'x'
219     for k in range(len(d)-1) :
220         if d[k] < x <= d[k+1]: break
221     return k
222
223 def COS (rb, y, bo):
224     # Returns the right or left bound of the type-reduced set through a center-of-sets
            type-reducer based on KM algorithm, being 'rb' the rulebase firings as a np.
            array, 'y' the output and 'bo' a boolean that determines which bound is going
            to be computed. Get 'output' and sort it in increasing order
225     Y = {n: y[n,1] if bo == 0 else y[n,0] for n in range(y.shape[0])}
226     # Initialise firing levels
227     F = [(rb[n,0] + rb[n,1]) * 0.5 for n in Y]
228     # Compute y
229     y1, y2 = sum([(Y[n] * F[n]) for n in Y]) / sum(F), 100
230     # Loop
231     for n in y:
232         # Check condition (y = y')
233         if abs(y2-y1) > 1e-6 : break
234         else                 : y1 = y2
235         # Find switch point
```

```
236         k = switch_point(Y, y1)
237         # Compute f
238         if bo == 0: F = [(rb[n,0]) if n <= k else (rb[n,1]) for n in Y]
239         else:       F = [(rb[n,1]) if n <= k else (rb[n,0]) for n in Y]
240         # Compute y'
241         y2 = float(sum([(Y[n] * F[n]) for n in Y])) / float(sum(F))
242     # Return
243     return y1
```

References

[1] Energy Networks Association EA Technologies. Assessing the impact of low carbon technologies on great britain's power distribution networks, 2012.

[2] Pranab J Baruah, Nicholas Eyre, Meysam Qadrdan, Modassar Chaudry, Simon Blainey, Jim W Hall, Nicholas Jenkins, and Martino Tran. Energy system impacts from heat and transport electrification. *Proceedings of the Institution of Civil Engineers-Energy*, 167(3):139–151, 2014.

[3] UK Department for Energy and Climate Change. Future potential for dsr in gb, 2015.

[4] Nicola Bui, Angelo P Castellani, Paolo Casari, and Michele Zorzi. The internet of energy: a web-enabled smart grid system. *IEEE Network*, 26(4):39–45, 2012.

[5] Ovidiu Vermesan, Peter Friess, Patrick Guillemin, Sergio Gusmeroli, Harald Sundmaeker, Alessandro Bassi, Ignacio Soler Jubert, Margaretha Mazura, Mark Harrison, M Eisenhauer, et al. Internet of things strategic research roadmap. *O. Vermesan, P. Friess, P. Guillemin, S. Gusmeroli, H. Sundmaeker, A. Bassi, et al., Internet of Things: Global Technological and Societal Trends*, 1:9–52, 2011.

[6] C.W. Gellings. Power to the people. *Power and Energy Magazine, IEEE*, 9(5):52–63, Sept 2011.

[7] H. Farhangi. A road map to integration: Perspectives on smart grid development. *Power and Energy Magazine, IEEE*, 12(3):52–66, May 2014.

[8] S. Rahman and O. Hazim. A generalized knowledge-based short-term load-forecasting technique. *IEEE Transactions on Power Systems*, 8(2):508–514, May 1993.

[9] Muhammad Qamar Raza and Abbas Khosravi. A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings. *Renewable and Sustainable Energy Reviews*, 50:1352 – 1372, 2015.

[10] D. Bunn and E. Farmer. *Comparative Models for Electrical Load Forecasting*. John Wiley and Sons, 1985.

[11] D.K. Ranaweera, G.G. Karady, and R.G. Farmer. Economic impact analysis of load forecasting. *Power Systems, IEEE Transactions on*, 12(3):1388–1392, Aug 1997.

[12] Heiko Hahn, Silja Meyer-Nieberg, and Stefan Pickl. Electric load forecasting methods: Tools for decision making. *European Journal of Operational Research*, 199(3):902–907, 2009.

[13] Einar Hope and Torstein Bye. Deregulation of electricity markets. *Economic and Political Weekly*, 40(50), Dec 2005.

[14] Severin Borenstein and James Bushnell. Electricity restructuring: Deregulation or reregulation? Competition policy center, working paper series, Competition Policy Center, Institute for Business and Economic Research, UC Berkeley, 2000.

[15] Torstein Bye and Einar Hope. Deregulation of electricity markets: the norwegian experience. *Economic and Political Weekly*, pages 5269–5278, 2005.

[16] Rafał Weron. Electricity price forecasting: A review of the state-of-the-art with a look into the future. *International Journal of Forecasting*, 30(4):1030–1081, 2014.

[17] B. F. Hobbs, S. Jitprapaikulsarn, S. Konda, V. Chankong, K. A. Loparo, and D. J. Maratukulam. Analysis of the value for unit commitment of improved load forecasts. *IEEE Transactions on Power Systems*, 14(4):1342–1348, Nov 1999.

[18] D.W. Bunn. Forecasting loads and prices in competitive power markets. *Proceedings of the IEEE*, 88(2):163–169, Feb 2000.

[19] Fahad Javed, Naveed Arshad, Fredrik Wallin, Iana Vassileva, and Erik Dahlquist. Forecasting for demand response in smart grids: An analysis on use

of anthropologic and structural data and short term multiple loads forecasting. *Applied Energy*, 96:150 – 160, 2012. Smart Grids.

[20] H. K. Alfares and M. Nazeeruddin. Electric load forecasting: literature survey and classification of methods. *International Journal of Systems Science,*, 33(1), 2002.

[21] James W Taylor and Patrick E McSharry. Short-term load forecasting methods: An evaluation based on european data. *Power Systems, IEEE Transactions on*, 22(4):2213–2219, 2007.

[22] I.S. Moghram and S. Rahman. Analysis and evaluation of five short-term load forecasting techniques. *Power Systems, IEEE Transactions on*, 4(4):1484–1491, Nov 1989.

[23] H.S. Hippert, C.E. Pedreira, and R.C. Souza. Neural networks for short-term load forecasting: a review and evaluation. *Power Systems, IEEE Transactions on*, 16(1):44–55, Feb 2001.

[24] V.H. Ferreira and A.P. Alves da Silva. Toward estimating autonomous neural network-based electric load forecasters. *Power Systems, IEEE Transactions on*, 22(4):1554–1562, Nov 2007.

[25] A. Khosravi, S. Nahavandi, D. Creighton, and D. Srinivasan. Interval type-2 fuzzy logic systems for load forecasting: A comparative study. *Power Systems, IEEE Transactions on*, 27(3):1274–1282, Aug 2012.

[26] A. Khosravi and S. Nahavandi. Load forecasting using interval type-2 fuzzy logic systems: Optimal type reduction. *Industrial Informatics, IEEE Transactions on*, 10(2):1055–1063, May 2014.

[27] S. Jurado, J. Peralta, A. Nebot, F. Mugica, and P. Cortez. Short-term electric load forecasting using computational intelligence methods. In *Fuzzy Systems (FUZZ), 2013 IEEE International Conference on*, pages 1–8, July 2013.

[28] Wei-Chiang Hong. Electric load forecasting by support vector model. *Applied Mathematical Modelling*, 33(5):2444 – 2454, 2009.

[29] K. Metaxiotis, A. Kagiannas, D. Askounis, and J. Psarras. Artificial intelligence in short term electric load forecasting: a state-of-the-art survey for the researcher. *Energy Conversion and Management*, 44(9):1525 – 1534, 2003.

[30] Elias Kyriakides and Marios Polycarpou. *Short Term Electric Load Forecasting: A Tutorial*, pages 391–418. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[31] S. Li, P. Wang, and L. Goel. A novel wavelet-based ensemble method for short-term load forecasting with hybrid neural networks and feature selection. *IEEE Transactions on Power Systems*, 31(3):1788–1798, May 2016.

[32] Jerry M. Mendel. *Uncertain Rule-Based Fuzzy Logic System: Introduction and New Directions*. Prentice Hall, 2001.

[33] N. Balta-Ozkan, T. Watson, P. Connor, C. Axon, L. Whitmarsh, R. Davidson, A. Spence, P. Baker, D. Xenias, L Cipcigan, and G. Taylor. Scenarios for the development of smart grids in the uk - synthesis report. *UK Research Energy Centre (UKERC)*, Feb. 2014.

[34] European Research Knowledge Center. Research challenges to increase the flexibility of power systems.

[35] Energy Networks Association. Active network management, good practice guide, 2015.

[36] Rodrigo Hidalgo, Chad Abbey, and Géza Joós. A review of active distribution networks enabling technologies. In *IEEE PES General Meeting*, pages 1–9. IEEE, 2010.

[37] Karim L Anaya and Michael G Pollitt. Experience with smarter commercial arrangements for distributed wind generation. *Energy Policy*, 71:52–62, 2014.

[38] Tao Xu and PC Taylor. Voltage control techniques for electrical distribution networks including distributed generation. *IFAC Proceedings Volumes*, 41(2):11967–11971, 2008.

[39] Guido Pepermans, Johan Driesen, Dries Haeseldonckx, Ronnie Belmans, and William D'haeseleer. Distributed generation: definition, benefits and issues. *Energy policy*, 33(6):787–798, 2005.

[40] Q Zhou and JW Bialek. Generation curtailment to manage voltage constraints in distribution networks. *IET Generation, Transmission & Distribution*, 1(3):492–498, 2007.

[41] Mats Larsson. *Coordinated voltage control in electric power systems*. Lund University, 2001.

[42] M Fila, D Reid, GA Taylor, P Lang, and MR Irving. Coordinated voltage control for active network management of distributed generation. In *2009 IEEE Power & Energy Society General Meeting*, pages 1–8. IEEE, 2009.

[43] EM Davidson, MJ Dolan, GW Ault, and SDJ McArthur. Aura-nms: An autonomous regional active network management system for edf energy and sp energy networks. In *IEEE PES General Meeting*, pages 1–6. IEEE, 2010.

[44] F Pilo, G Pisano, and GG Soma. Advanced dms to manage active distribution networks. In *PowerTech, 2009 IEEE Bucharest*, pages 1–8. IEEE, 2009.

[45] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26(1):12–19, Feb 2011.

[46] Pengfei Wang, Daniel H Liang, Jialiang Yi, Pádraig F Lyons, Peter J Davison, and Philip C Taylor. Integrating electrical energy storage into coordinated voltage control schemes for distribution networks. *IEEE Transactions on Smart Grid*, 5(2):1018–1032, 2014.

[47] Michael J Krok and Sahika Genc. A coordinated optimization approach to volt/var control for large power distribution networks. In *Proceedings of the 2011 American Control Conference*, pages 1145–1150. IEEE, 2011.

[48] Sergi Cabré Ramos. Optimization of the operation of a distribution network with distributed generation using genetic algorithm. 2014.

[49] Vladimiro Miranda and Nuno Fonseca. New evolutionary particle swarm algorithm (epso) applied to voltage/var control. In *Proceedings of the 14th power systems computation conference (PSCC)*, pages 1–6, 2002.

[50] J Sugimoto, R Yokoyama, Y Fukuyama, VVR Silva, and H Sasaki. Coordinated allocation and control of voltage regulators based on reactive tabu search. In *Power Tech, 2005 IEEE Russia*, pages 1–6. IEEE, 2005.

[51] Qilian Liang and Jerry M. Mendel. Interval type-2 fuzzy logic systems: theory and design. *IEEE T. Fuzzy Systems*, 8(5):535–550, 2000.

[52] AG Madureira and JA Pecas Lopes. Coordinated voltage support in distribution networks with distributed generation and microgrids. *IET Renewable Power Generation*, 3(4):439–454, 2009.

[53] G.N. Taranto, A.B. Marques, and D.M. Falcao. Coordinated voltage control using fuzzy logic. In *Power Engineering Society Summer Meeting, 2002 IEEE*, volume 3, pages 1314–1317 vol.3, Jul. 2002.

[54] A Sajadi, HE Farag, P Biczel, and EF El-Saadany. Voltage regulation based on fuzzy multi-agent control scheme in smart grids. In *Energytech, 2012 IEEE*, pages 1–5. IEEE, 2012.

[55] V. Miranda, A. Moreira, and J. Pereira. An improved fuzzy inference system for voltage/var control. *Power Systems, IEEE Transactions on*, 22(4):2013–2020, Nov. 2007.

[56] D.H. Spatti, I.N. da Silva, W.F. Usida, and R.A. Flauzino. Real-time voltage regulation in power distribution system using fuzzy control. *Power Delivery, IEEE Transactions on*, 25(2):1112–1123, Apr. 2010.

[57] V. Loia, A. Vaccaro, and K. Vaisakh. A self organizing architecture based on cooperative fuzzy agents for smart grid voltage control. *Industrial Informatics, IEEE Transactions on*, PP(99):1–1.

[58] P.H. Nguyen, J. M A Myrzik, and W.L. Kling. Coordination of voltage regulation in active networks. In *Transmission and Distribution Conference and Exposition, 2008. T x00026;D. IEEE/PES*, pages 1–6, 2008.

[59] Hani Hagras. A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. *IEEE Transactions on Fuzzy Systems*, 12(4):524–539, 2004.

[60] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. *Information Sciences*, 1:119–249, 1975.

[61] Yong-Hua Song and Allan T Johns. Applications of fuzzy logic in power systems. i. general introduction to fuzzy logic. *Power Engineering Journal*, 11(5):219–222, 1997.

[62] Mohammed E El-Hawary. *Electric power applications of fuzzy systems*. Wiley-IEEE Press, 1998.

[63] R.C. Bansal. Bibliography on the fuzzy set theory applications in power systems (1994-2001). *Power Systems, IEEE Transactions on*, 18(4):1291–1299, Nov 2003.

[64] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[65] J. Yen and Liang Wang. Application of statistical information criteria for optimal fuzzy model construction. *Fuzzy Systems, IEEE Transactions on*, 6(3):362–372, Aug 1998.

[66] Lotfi A Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy sets and systems*, 100:9–34, 1999.

[67] J.M. Mendel, R.I. John, and Feilong Liu. Interval type-2 fuzzy logic systems made simple. *Fuzzy Systems, IEEE Transactions on*, 14(6):808–821, 2006.

[68] H. Hagras. Type-2 flcs: A new generation of fuzzy controllers. *Comp. Intell. Mag.*, 2(1):30–43, February 2007.

[69] D. Wu and W.-W. Tan. Type-2 fls modeling capability analysis. In *Fuzzy Systems, 2005. FUZZ '05. The 14th IEEE International Conference on*, pages 242–247, 2005.

[70] Christian Wagner and Hani Hagras. Toward general type-2 fuzzy logic systems based on zslices. *IEEE T. Fuzzy Systems*, 18(4):637–660, 2010.

[71] Dongrui Wu Woei Wan Tan. A simplified type-2 fuzzy logic controller for real-time control. $\{ISA\}$ *Transactions*, 45(4):503 – 516, 2006.

[72] Hani Hagras and Christian Wagner. Towards the wide spread use of type-2 fuzzy logic systems in real world applications. *IEEE Comp. Int. Mag.*, 7(3):14–24, 2012.

[73] Hani Hagras and Christian Wagner. Introduction to interval type-2 fuzzy logic controllers—towards better uncertainty handling in real world applications. *IEEE Systems, Man and Cybernetics eNewsletter*, 27, 2009.

[74] Jerry M Mendel, Hani Hagras, and Robert I John. Standard background material about interval type-2 fuzzy logic systems that can be used by all authors, 2006.

[75] Dongrui Wu. A brief tutorial on interval type-2 fuzzy sets and systems. *University of Southern California, USA2012*, 2010.

[76] Ebrahim H Mamdani and Sedrak Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International journal of man-machine studies*, 7(1):1–13, 1975.

[77] Michio Sugeno and Takahiro Yasukawa. A fuzzy-logic-based approach to qualitative modeling. *IEEE Transactions on fuzzy systems*, 1(1):7–31, 1993.

[78] Ben Coppin. *Artificial intelligence illuminated*. Jones & Bartlett Learning, 2004.

[79] D. Wu. Twelve considerations in choosing between gaussian and trapezoidal membership functions in interval type-2 fuzzy logic controllers. In *2012 IEEE International Conference on Fuzzy Systems*, pages 1–8, June 2012.

[80] Roberto Sepúlveda, Oscar Castillo, Patricia Melin, Antonio Rodríguez-Díaz, and Oscar Montiel. Experimental study of intelligent controllers under uncertainty using type-1 and type-2 fuzzy logic. *Information Sciences*, 177(10):2023–2048, 2007.

[81] Nilesh N. Karnik and Jerry M. Mendel. Centroid of a type-2 fuzzy set. *Information Sciences*, 132(1–4):195 – 220, 2001.

[82] Hongwei Wu and Jerry M Mendel. Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems. *IEEE Transactions on fuzzy systems*, 10(5):622–639, 2002.

[83] Dongrui Wu and Jerry M Mendel. On the continuity of type-1 and interval type-2 fuzzy logic systems. *IEEE Transactions on Fuzzy Systems*, 19(1):179–192, 2011.

[84] Dongrui Wu. Approaches for reducing the computational cost of interval type-2 fuzzy logic systems: Overview and comparisons. *Fuzzy Systems, IEEE Transactions on*, 21(1):80–99, 2013.

[85] Maowen Nie and Woei Wan Tan. Towards an efficient type-reduction method for interval type-2 fuzzy logic systems. In *Fuzzy Systems, 2008. FUZZ-IEEE 2008. (IEEE World Congress on Computational Intelligence). IEEE International Conference on*, pages 1425–1432, June 2008.

[86] J. Yen. Fuzzy logic-a modern perspective. *Knowledge and Data Engineering, IEEE Transactions on*, 11(1):153–165, Jan 1999.

[87] S. M. Viegas, J. L.and Vieira, R. Melicio, V. M. F. Mendes, and J. M. C. Sousa. *GA-ANN Short-Term Electricity Load Forecasting*, pages 485–493. Springer International Publishing, Berlin, Heidelberg, 2016.

[88] Nantian Huang, Zhiqiang Hu, Guowei Cai, and Dongfeng Yang. Short term electrical load forecasting using mutual information based feature selection with generalized minimum-redundancy and maximum-relevance criteria. *Entropy*, 18(9):330, 2016.

[89] Mashud Rana, Irena Koprinska, and Abbas Khosravi. Feature selection for interval forecasting of electricity demand time series data. In Petia Koprinkova-Hristova, Valeri Mladenov, and Nikola K. Kasabov, editors, *Artificial Neural Networks*, volume 4 of *Springer Series in Bio-/Neuroinformatics*, pages 445–462. Springer International Publishing, 2015.

[90] M. A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

[91] Yonghan Feng and Sarah M. Ryan. Day-ahead hourly electricity load modeling by functional regression. *Applied Energy*, 170:455 – 465, 2016.

[92] Bartosz Uniejewski, Jakub Nowotarski, and Rafał Weron. Automated variable

selection and shrinkage for day-ahead electricity price forecasting. *Energies*, 9(8):621, 2016.

[93] Robin Genuer, Jean-Michel Poggi, and Christine Tuleau-Malot. Variable selection using random forests. *Pattern Recognition Letters*, 31(14):2225 – 2236, 2010.

[94] Gavin Brown. A new perspective for information theoretic feature selection. In David V. Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS-09)*, volume 5, pages 49–56. Journal of Machine Learning Research - Proceedings Track, 2009.

[95] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Phys. Rev. E*, 69:066138, Jun 2004.

[96] Greg Ver Steeg and Aram Galstyan. Inferring predictive links in social media using content transfer. *CoRR*, abs/1208.4475, 2012.

[97] Robert J. May, Holger R. Maier, Graeme C. Dandy, and T.M.K. Gayani Fernando. Non-linear variable selection for artificial neural networks using partial mutual information. *Environmental Modelling Software*, 23(10–11):1312 – 1326, 2008.

[98] Stephen L Chiu. Fuzzy model identification based on cluster estimation. *Journal of Intelligent & fuzzy systems*, 2(3):267–278, 1994.

[99] Agus Priyono, Muhammad Ridwan, Ahmad Jais Alias, Riza Atiq OK Rahmat, Azmi Hassan, and Mohd Alauddin Mohd Ali. Generation of fuzzy rules with subtractive clustering. *Jurnal Teknologi*, 43(1):143–153, 2012.

[100] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[101] S. Beri and K. Kaur. Hybrid framework for dbscan algorithm using fuzzy logic. In *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, pages 383–387, Feb 2015.

[102] Pablo Moscato. On evolution, search, optimization, genetic algorithms and martial arts - towards memetic algorithms, 1989.

[103] D. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems*, 3(5):493–530, 1989.

[104] Ferrante Neri and Carlos Cotta. Memetic algorithms and memetic computing optimization: A literature review. *Swarm and Evolutionary Computation*, 2:1–14, 2012.

[105] Jaume Bacardit and Josep Maria Garrell. Bloat control and generalization pressure using the minimum description length principle for a pittsburgh approach learning classifier system. In *Learning Classifier Systems*, volume 4399 of *Lecture Notes in Computer Science*, pages 59–79. Springer Berlin Heidelberg, 2007.

[106] Frank Hoffmann and Gerd Pfister. Evolutionary design of a fuzzy knowledge base for a mobile robot. *International Journal of Approximate Reasoning*, 17(4):447–469, 1997.

[107] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[108] Adrian S. Lewis and Michael L. Overton. Nonsmooth optimization via quasi-newton methods. *Mathematical Programming*, 141(1):135–163, 2013.

[109] Fernando Pérez-Cruz. Estimation of information theoretic measures for continuous random variables. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1257–1264. Curran Associates, Inc., 2009.

[110] B. Sareni and L. Krahenbuhl. Fitness sharing and niching methods revisited. *Evolutionary Computation, IEEE Transactions on*, 2(3):97–106, Sep 1998.

[111] Jose Joaquin Aguilera, Manuel Chica, Maria Jose del Jesus, and Francisco Herrera. Niching genetic feature selection algorithms applied to the design of fuzzy rule-based classification systems. In *2007 IEEE International Fuzzy Systems Conference*, pages 1–6. IEEE, 2007.

[112] Joshua Knowles and David Corne. Memetic algorithms for multiobjective optimization: issues, methods and prospects. In *Recent advances in memetic algorithms*, pages 313–352. Springer, 2005.

[113] Yew-Soon Ong, Meng Hiot Lim, and Xianshun Chen. Research frontier-memetic computation—past, present & future. *IEEE Computational Intelligence Magazine*, 5(2):24, 2010.

[114] Martin Casdagli. A dynamical systems approach to modeling input-output systems. In *SFI Studies in the Sciences of Complexity*, volume 12, pages 265–281. Addison-Wesley, 1992.

[115] Ho Jae Lee, Jin Bae Park, and Young Hoon Joo. Fuzzy model identification using a hybrid mga scheme with application to chaotic system modeling. In *Integration of Fuzzy Logic and Chaos Theory*, volume 187 of *Studies in Fuzziness and Soft Computing*, pages 81–97. Springer Berlin Heidelberg, 2006.

[116] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53 – 65, 1987.

[117] Wes McKinney. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. O'Reilly, Beijing, first edition, 2013.

[118] Christian Rehtanz. *Autonomous systems and intelligent agents in power system control and operation*. Springer Science & Business Media, 2003.

[119] A. Elmitwally, M. Elsaid, M. Elgamal, and Z. Chen. A fuzzy-multiagent self-healing scheme for a distribution system with distributed generations. *IEEE Transactions on Power Systems*, 30(5):2612–2622, Sept 2015.

[120] P. C. Taylor, T. Xu, N. S. Wade, M. Prodanovic, R. Silversides, T. Green, E. M. Davidson, and S. McArthur. Distributed voltage control in aura-nms. In *IEEE PES General Meeting*, pages 1–7, July 2010.

[121] Northern Powergrid. Customer led network revolution, 2015.

[122] M. A. Hashim, J. Jaafar, and S. M. Taib. New forecasting model using type-2 fuzzy multivariate time series. In *2011 National Postgraduate Conference*, pages 1–4, Sept 2011.

[123] I. Riaño and O. E. Perdomo. Electricity price forecasting using a fuzzy system tuned with a differential evolution algorithm. In *2015 IEEE PES Innovative Smart Grid Technologies Latin America (ISGT LATAM)*, pages 792–795, Oct 2015.

[124] P. Pinson and M. O'Malley. Foreword for the special section on wind and solar energy: Uncovering and accommodating their impacts on electricity markets. *Power Systems, IEEE Transactions on*, 30(3):1557–1559, May 2015.