ROMA TRE
UNIVERSITÀ DEGLI STUDI

Roma Tre University
Ph.D. in Computer Science and Engineering

Application of Computational Intelligence to Energy Systems

Matteo De Felice

Application of Computational Intelligence to Energy Systems

A thesis presented by
Matteo De Felice
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Engineering

Roma Tre University
Dept. of Informatics and Automation

March 2011

“*Everything should be made as simple as possible, but not simpler*”
— Albert Einstein

“*Frustra fit per plura quod potest fieri per pauciora* (it is useless to do with more what can be done with less) ” — William of Ockham

Acknowledgments

Although this work is the result of a personal and professional journey started in 2007, it would have been, if not impossible, at least very difficult (and less pleasant), without the help and the support of many people.

First of all, I would like to thank my Ph.D. advisor, Prof. Stefano Panzieri, for his advices and assistance during these years.

My gratitude goes also to my referees, Prof. Xin Yao and Prof. Emilio Corchado, for their precious suggestions.

I am very grateful to Mauro Annunziato and Ilaria Bertini, who made possible this collaboration between ENEA and the University of Rome "Roma Tre" and because they have contributed to create the best environment for my research. I would like to thank also my ENEA colleagues: Francesco Ceravolo, Biagio Di Pietra, Francesca Margiotta, Alessandro Pannicelli, Stefano Pizzuti and Giovanni Puglisi, for their continued support and knowledge during these years.

Thanks to Francesco Corman, Maurizio Di Rocco, Guido Drovandi and Sandro Meloni for their enthusiasm and for a lot of interesting discussions on our respective research topics (and off-topics).

I reserve special thanks for my family, in all these years (and not only) their encouragement and support have been a constant, an extremely precious one.

Finally, I'd like to thank Manuela, she has always been my first supporter and advisor, providing me the enthusiasm sometimes I didn't have to continue my work.

Contents

CHAPTER 1

Introduction

Under the definition of *computational intelligence techniques* falls an emerging family of problem-solving and problem-stating methods that attempt to model the intelligence that we can observe in nature.

Andries Engelbrecht, in his book [68], define the computational intelligence as:

> [...] the study of adaptive mechanisms to enable or facilitate intelligent behavior in complex and changing environments.

This definition gives emphasis to adaptation skills which are based on discover, learn and generalize knowledge, all abilities fundamental for the animals to ensure survival and reproduction.

The term Computational Intelligence is strictly related to the more general *soft computing*, a field of computer science characterized by the use of inexact solutions for computationally-hard problems [129]. This kind of approach takes into consideration the uncertainty, approximation and tolerance to imprecisions exploiting them for dealing with problems, in contrast to the so-called *hard-computing* which strive for mathematical accuracy and preciseness. The model of soft computing is the human mind, and the main paradigm is the machine learning/computational learning one: the possibility to learn from observation in order to recognize and generalize new situations (or data). The first paper with the definition of soft computing appeared in 1965 [235, 236] but the methods it includes are earlier, like neural networks which first work was published in 1958. Despite the two terms, computational intelligence and soft computing, both include the same important methodologies (e.g. neural networks and fuzzy logic) the former puts emphasis on the 'intelligence' of these methods while the latter underlines that these methods are less rigorous, and so 'softer', than hard computing.

Computational Intelligence algorithms, and the soft computing field itself, with deductive and case-based reasoning and a wide variety of learning systems, may be considered a sub-branch of Artificial Intelligence (AI) which focus, as the name itself explains, to the artificial definition of 'intelligence'. The debate about the definition of intelligence or about the relation between computation/computing and intelligence itself are outside the topic of this dissertation, but it's clear that the definitions and the labels on this wide area are not simple, and probably not fundamental.

It should be clear that computational intelligence paradigm is a better model of the intelligence we can found in nature than hard computing, in fact precision and rigorous calculations are not terms usually referred to human (or more in general, animal) computation. On the other hand, fast calculations and learning are easily associated to the human brain: a man is able to learn, generalize, recognize and take decisions on new situations never faced and observed before and he could process thousands of information of different typologies (visual, acoustic, factual etc) in a short time with a strong tolerance to errors, noise and disturbs: all abilities highly desirable for a computer system in a lot of fields (e.g. robotics or speech recognition).

However, searching a trade-off between precision/accuracy and costs/time is necessary if we want to build models and systems with the same basic features of human brain. In real-world problems, we often need to accept suboptimal solutions in order to make acceptable the computational effort, i.e. the time for a solution, defining what is "good enough" for the specific problem. The search of this trade-off for a problem is the first task for an engineer designing an algorithm for a specific application: he can choose on a wide range of methods, from a trivial random sampling (low accuracy and low computational cost) to an exhaustive search (very high accuracy and very high computational cost). All the heuristics, and computational intelligence methods too, are normally in between these two extremes.

Ho [104] gives an interesting example to explain the advantages of a "good enough" philosophy, against a "best for sure" one. Let us consider a problem consisting of of choosing in a search space of size D a sample in the top n with N random samples. With $D = 10^9$ (a small size search space for real applications) and $N = 1000$ the probability to choose the best sample ($n = 1$) is $1 - (1 - 1/D)^N = 10^{-6}$, a very low probability. This value increases to 0.01 if we consider "good enough" a value in the top $n = 10000$, i.e. widening our

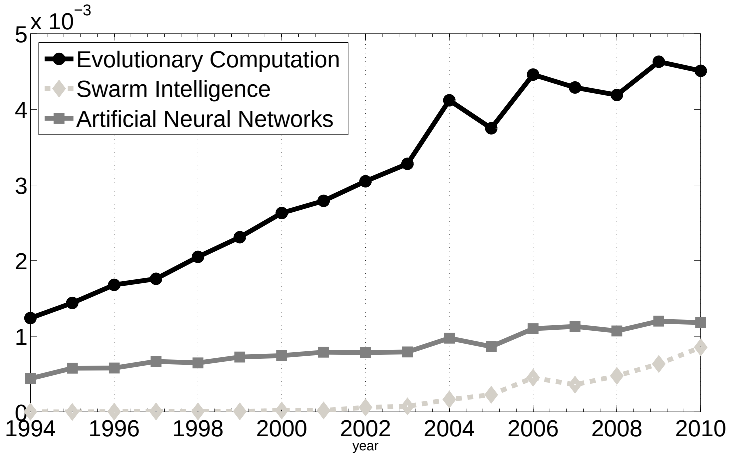


Figure 1.1: Number of paper published from 1994 to 2010 in different topics

concept of "good".

Taxonomy of Computational Intelligence Methods

In Figure 1.1 is shown the ratio between number of paper about a topic and all the papers published[1] in Evolutionary Computation, Swarm Intelligence, and Artificial Neural Networks. Data is collected from Web of Science[2]. All the three considered topics show an increasing trend, more evident for Evolutionary Computation and Swarm Intelligence, research fields more recent than neural networks.

Each computational intelligence algorithm has its origin in biological systems, in the following list is presented a brief summary of the natural system which inspired the methodology and the first year of appearance in scientific literature.

Neural Networks (NNs) models human brain and more in general, neural systems. The first paper of McCulloch and Pitts [164] was published

[1]Data are updated at January 2010 and only the Computer Science and Technology section is considered.

[2]Web of Science (`http://scientific.thomson.com/products/wos/`) by Thomson Reuters is an academic service which database covers at January 2010 about 90 millions of scientific papers.

in 1943, proposing a model for the biological neurons, but the first neuron computing model, called *perceptron*, was proposed fifteen years later by Rosenblatt [196]

Evolutionary Computation (EC) models natural evolution. Evolution Strategies (ES) was developed by Rechenberg in the 60s [192], Evolutionary Programming (EP) by Fogel in 1962 [77] and the main work about Genetic Algorithms (GAs) is considered the book of John Holland [106] in 1975

Swarm Intelligence (SI) is inspired by the behaviour of swarms or colonies. The first swarm algorithm, Ant Colony Optimization (ACO), was proposed by Dorigo in 1992 [62]. In 1995 Particle Swarm Optimization (PSO), was published by Eberhart and Kennedy [130].

Artificial Immune Systems (AIS) models the human immune system. The first theoretical work about Clonal Selection principle was published by Burnet [30] in 1959 but the first algorithm was presented by Forrest [82] in 1994.

This list can not be considered exhaustive but it's an attempt of outline the main algorithms applied in this thesis. Even though the separations between each family are clear, there are many hybrid approaches that try to mix the features of techniques in order to find the optimal solution for a problem, e.g. using evolutionary computation algorithms to find parameters of neural networks.

Complex Methods for Complex Problems

Computational intelligence (CI) algorithms are commonly used for different purposes, we can outline two main categories of application: learning/modelling and optimization. Despite these two problem are partially overlapped, in fact a learning problem can be formulated in an optimization form, the CI algorithms have been designed for particular applications and they have gained popularity especially where particular good results were obtained.

Computational Intelligence techniques presented more or less important limitations that limited in some cases their diffusion. The main drawback is the high computational requirements for optimization applications, in fact CI algorithms are generally stochastic methods and therefore need an high number of iterations in order to obtain good and feasible solutions, a number generally higher than traditional methods in simple problems. Moreover, usually a large number of parameters are needed to tune these algorithms and in some cases (e.g. back-propagation algorithm) there are in literature no more than empirical tests and best practises, leading to the common execution of a time-consuming trial-and-error to obtain the optimal parameters for a specific problem. Algorithms parameter settings are always problem-dependent and the algorithms performances are normally very sensitive to them. In learning and modelization problems, given that CI methods are generally data-driven, the main shortcoming is that to obtain a good model the data provided has to be a good representation of the problem space itself, i.e. data should contain information spread evenly throughout the entire working range of the system.

Despite limitations and drawback, CI algorithms are commonly used, especially where an analytical model of the problem is impossible or too expensive, or when due to non-linearity of the problem, modelling accuracy of traditional techniques doesn't permit to obtain practical good solutions. When there is not enough knowledge or expertise to design problem-specific algorithms, randomized heuristics and CI approaches can perform effectively. Of course incorporate problem-specific knowledge into them might improve their performance.

Commonly, where an accurate mathematical modelling of the problem is possible, the application of traditional algorithms, which often implies calculation of derivatives or gradient, certainly leads to the best results, i.e. in function optimization if the second derivative is present a Newton's Method like Levenberg-Marquardt, which is fastest compared to an Evolutionary Computation method. In other cases, CI techniques, although usually not providing mathematically optimal solutions, can lead to "good enough" solutions, sometimes near the optimal ones.

Another major advantage of CI algorithms is that they usually permit to lead with large-scale optimization problems where classical techniques, like branch-and-bound and dynamic programming, take an unreasonable amount of time to find an optimal solutions, or in presence of noisy data or high number of variables.

We can draw the conclusion that CI may cope effectively with the complexity and the difficulty of real-world problem. What does "complex" means? High-dimensionality, problem parameters strongly dependent, lack of information, noisy and corrupted data, real-time requirements etc. Even with these conditions CI algorithms provided good performances in a wide variety of application fields:

- Industrial fault detection

- Computer networks intrusion detection

- Robotic control and planning
- Forecasting
- System identification
- Medical diagnosis
- Industrial scheduling and vehicle routing
- Financial modelling and risk estimation
- Software engineering
- Signal and image processing
- Sound and speech recognition
- Networks routing
- Process control and anomaly detection

This list is not exhaustive, however reviews for real-world applications can be found in scientific literature, e.g. in [166] for neural networks and in [53] for AIS.

In the last years a particular engineering field became important and a particularly interesting application field: energy systems. All the problems relative to energy like energy management of facilities and buildings, optimal design of energy systems, scheduling, load forecasting, planning, became more important with the increase of needs of energy efficiency, due to fuels' higher prices and better sensibility of public opinion to pollution and consumptions. CI approaches are well suited to deal with such problems that usually are complex considering non-convex and non-differentiable functions, noisy and disturbed data, multiple local optima, constrained domains and more. This field has two main features which makes it very interesting: the urgency of solutions to problems of public interest and the high degree of complexity of involved systems and subsystems which involves knowledge of very different areas (thermodynamics, computer science, ergonomy, social sciences, economics, etc). Given that many energy-related problems involve pollutant emissions control and fuel consumption efficiency (i.e. costs), it's clear the urgency of these problems, urgency which caused increasing interest from public opinion and not only from scientific fields.

There are several books about the application of CI algorithms and other heuristics to energy problems, for a good example see [149, 242]

ORGANIZATION OF THE WORK

This thesis is the resulting work of three years spent working at ENEA (Italian Energy, New Technology and Sustainable Economic Development Agency) on various projects concerning different aspects of energy systems: from the temperature modelling with the aim of improving energy building simulation to the optimization of a combined cycle power plant.

This work is organized in four chapters: Chapter 2 reviews and describe applications of computational intelligence algorithms to building energy systems and the Chapters 3, 4, and 5 describe three different real-world applications.

Chapter 3 describes the application of neural networks to temperature modelling. The problem of modelling hourly ambient temperatures of any Italian location, given its geographical coordinates, is described and an hybrid approach based on computational intelligence techniques in order to estimate monthly and daily temperature is proposed. Back-propagation (BP) algorithm and a Genetic Algorithm (GA) are combined to train effectively neural networks in such a way that the BP algorithm initialises a few individuals of the GA's population. Experiments concerned monthly temperature estimation of unknown places and daily temperature estimation for thermal load computation. This work was published in [23].

In Chapter 4 is described an application of fuzzy-logic and evolutionary computation to the optimization of the start-up phase of a combined cycle power plant. The process with fuzzy sets over the process variables starting from experts' knowledge in order to get the needed cost function for the Genetic Algorithm (GA) used to obtain the optimal parameters. Due to the obvious impossibility to test the resulting inputs on the real plant a complex software simulator is used to evaluate the performance of the solutions. In order to reduce the computational load of the whole procedure for the genetic algorithm a novel fitness approximation technique is implemented, achieving a cutting by 98% the number of fitness evaluations, i.e. software simulator runs with respect to a Genetic Algorithm without fitness approximation. Moreover, a multi-objective approach has been proposed and applied with interesting results. This work was published in [22].

Finally, in Chapter 5 is proposed a new approach for short-term load forecasting based on neural networks ensembling methods. A comparison between traditional statistical linear seasonal model and NN-based models has been performed on real building load data, considering the utilisation of external data such as the day of the week and building occupancy data. The selected models have been compared to the prediction of hourly demand for the electric power up to 24 hours for a testing week. This paper has been published in [54].

This work is concluded in Chapter 6 with a discussion on the results presented and some indications about the future work.

CHAPTER 2

Overview of Applications in Building Energy Systems

"Complex problems have simple,
easy to understand wrong answers."
Grossman's Law

Organization of the Chapter

In this Chapter we present a survey covering the area of computational intelligence algorithms to optimization, control and modelling in building energy systems.

1. A brief introduction to energy-related problems is given in Section 2.1
2. Section 2.2 describes the methodologies covered in this survey
3. Section 2.3 examines several applications of computational intelligence to optimization, modelling, control and forecasting problems related to building energy systems.

2.1 Introduction to Energy-related Problems

We can try to define energy-related problems as all the problems involving the extraction, collection and utilisation of energy resources to satisfy some specific needs. This definition definitely includes many different fields of engineering and science from civil engineering to mathematics, from computer science to climatology. This chapter will focus on building energy systems, an area, which due to its complexity and wide variety, has high potential on the application of computational intelligence approaches, which application may be seen as an alternative to conventional methods when the latter don't provide effective solutions.

Algorithms usually employed to cope with all the problems related to building energy systems require the solution of complex differential equations. In this case, at first a physic model is needed and in the majority of cases only a simplified model is available due to the lack of information (and time) for a more realistic model. This kind of approach requires an high computational power and thus a considerable amount of time to have accurate results. Moreover, data from building energy systems are often 'noisy' and incomplete, two characteristics that makes the problems unsuitable for classical methods. To be able to model the behaviour of building energy systems, it is necessary to consider non-linear multivariate (and noisy) systems with an high number of relations between the different parts of the system and the environment (e.g. weather information).

Thus, although analytical models have been useful to study such systems, with the increasing computational power of the last decades, numerical methods have became much more attractive than analytical solutions with the advantage of handling more complex and realistic situations.

Many of the problems experienced in buildings energy systems appears to be most suited for computational intelligence approach. In fact, all these approaches strongly rely on system models (software models or differential equations), which due to the complexity of the physical original systems show different degrees of approximation, and data collected from sensors.

An overview of artificial intelligence methods applied to building energy systems was published by Krarti [138], which gives a focus on forecasting and modeling tasks. Definitely more specialised is the work proposed by Gosselin et al. [90] which reviews all the applications of Genetic Algorithms to heat transfer problems, thus considering common HVAC (Heating, Ventilating, and Air Conditioning) related optimisation problems.

On the basis of typology of problems we can define three main areas:

1. Forecasting/modelling
2. Control
3. Optimization

Forecasting, the process of estimation of future situations, is a fundamental task for planning, management, scheduling and for all the operations involving the need of accurate information about the system state at a precise instant t. Modelling instead, is a complementary task respect to forecasting because an accurate representation of a system, where possible, is important to predict its behaviour after specific inputs. In many cases the modellization of a system can be performed with more accuracy with a statistical bottom-up strategy (e.g. neural networks) respect to top-down approaches, which require detailed information about all the parameters involved.

Control is the task of manage and regulate the behaviour of a system to achieve a desired output. Possibly it involves modelling and forecasting in order to get an estimation of the outputs up against controlled inputs.

Optimization is the fundamental process of find and choose the best alternative with respect to one (or more) error measures, fulfilling physical constraints.

2.2 Computational Intelligence Methods

For this survey has been considered all the scientific literature published on the last fifteen years related to the application of computational intelligence techniques of building energy systems. In Table 2.1 all the selected papers has been classified by their main problem area: forecasting/modelling, optimisation and control.

2.2.1 Neural Networks

Neural Networks are evidently a common tool for energy-related problems: their ability to find relationships between observed data and to approximate, often with a satisfying precision, complex physical systems are in fact critical for the application to building energy systems. For this reason we can find neural networks used in a wide variety of approaches, they are common for forecasting tasks and they are often used as approximated models for control tasks when an estimation of the behaviour of the controlled system is required.

The most common typology of neural network is the feed-forward (FF) one, the name refers to the fact that the information flows only in one direction, from the input to the output nodes, without loops. Multilayer Perceptrons (MLPs) are most popular kind of neural networks of this typology. MLPs have at least three layers, one input layer, one output layer and a variable number of hidden layers. This kind of neural networks is used for several reasons, the first is probably the easiness of implementation (there are many available in

Table 2.1: Buildings survey

Field	Refs.
Optimization	
Efficient building design	[34, 230, 35, 127, 160, 224, 244]
HVAC System design	[78, 142, 128]
HVAC Operation settings	[58, 108, 45, 158, 176, 41, 38, 40, 150, 49, 184, 240, 241]
Model identification	[221, 239, 177]
Control	
HVAC Systems control	[232, 35, 79, 133, 10, 11, 20, 154, 170, 171, 223, 172]
Building Lighting	[93]
Intelligent Buildings	[173, 157]
Forecasting and Modeling	
Air flow forecasting	[13]
Energy forecasting	[122, 124, 14, 168, 16, 21, 60, 87, 233, 15, 180, 66, 153, 152, 61, 229, 143]
Prediction of Natural Lighting Levels	[48]
Thermal comfort index modelling	[12]
Prediction of indoor temperature	[91, 159, 165, 197]

commercial and open source software, see Appendix A) and second, the existence of an efficient training algorithm, the back-propagation (BP) algorithm. For this reason, MLPs are sometimes called back-propagation neural networks.

The BP algorithm performs a gradient descent in network weights space according to the error function and since its invention it has been extended with additional features, such as an adaptive learning rate (a parameter which influences the amplitude of changes of the weights) and the introduction of a momentum term for the weight changes formula (the delta rule). The problem of finding the best weights for a neural network can be defined as a classical numerical optimization problem, thus Newton and quasi-Newton methods can be applied. A commonly used algorithm is the Levenberg-Marquartd method [151] which exists also in its Hessian-approximated form.

Another frequent feed-forward neural network typology is the Radial-Basis Function (RBF) network, where into the hidden layer there is a set of radial-basis functions (commonly Gaussian functions) which apply a nonlinear transformation from the input to the hidden high-dimensional space. This structure has a strong theoretical justification into the Cover's theorem [51] on the separability of patterns and it has a major advantage on its easiness of implementation. Generalized Regression Neural Networks (GRNNs) are similar to RBF networks, they have in fact a Gaussian function as hidden function for each training instance which returns a value indicating the degree of similarity between the input vector and the particular training instance.

Differently from feed-forward networks, recurrent networks have at least one feedback loop and various models have been presented during the years (Elman networks, Jordan networks, Hopfield networks, etc). Although this kind of network is able to approximate dynamical systems (see [208]), there is an evident problem in applying training algorithms based on descent methods (like backpropagation) and thus various alternative methods (often very computationally expensive) have been proposed during the years.

An hybrid model between neural networks and fuzzy logic is called ANFIS (Adaptive Network Based Fuzzy Inference System). This approach allows a mapping between input and outputs using a fuzzy inference system and it has been used successfully in many identification and control applications although its implementation results particularly time consuming.

A recent survey can be found in the work by Kalogirou [123] which review mainly the applications of neural networks on prediction and estimation.

In Table 2.2 all the NN-based implementations in selected papers are classified by the neural network typology and further information about training algorithm and structure are provided.

2.2.2 Evolutionary Computation

As evolution is the natural process where the best (fittest) organism tends to survive in a competitive and changing environment, Evolutionary Computation (EC) mimics this process "evolving" a set of solutions (population), creating better solutions generations after generations through the genetic operators of mutation and reproduction. The most common Evolutionary Algorithm is the Genetic Algorithm (GA), which model was proposed in the '50s but made popular by the extensive work of John Holland [106].

GA performs a stochastic search for an optimal solution, given a cost function called fitness function, though the solution space. Originally GA was in-

Table 2.2: Neural networks models. Abbreviations used for the training algorithms (where specified) are the following: simple backpropagation (BP), backpropagation with momentum (BPM), Levenberg-Marquardt (LM), Scaled-Conjugate Gradient (SCG)

Neural network typology	Reference with number of layers and training algorithm
MLP	[121, 223, 165, 233, 66, 61, 160, 170](3-layers, LM), [10] (3/4-layers, BPM),[124, 125, 122, 87] (3-layers, BPM), [171, 141, 229] (3-layers, BP), [14] (3/5-layers, QuickProp), [45] (4-layers, LM), [168, 180, 153] (3-layers, BP), [11] (4-layers, BPM), [16] (3-layers, BP and Resilient BP), [60, 12] (4-layers, BP), [13] (3-layers, SCG and LM)
RBF	[197] (LM and K-Means clustering), [153], [93]
GRNN	[20, 21, 153]
Recurrent networks	[124, 125] (3-layers)
ANFIS	[158, 152, 157]

tended to have a discrete solution space, where each solution was encoded as a binary string, but nowadays is common to consider GA also with a real-valued encoding where the solution is represented with a vector.

The search process is made up of three main steps: selection, reproduction and mutation. The first step has the objective of emphasize better solutions, the second step to combine them to create offspring with a better fitness, and the last one, mutation, has the aim to introduce new solution components adding diversity into the population. Evolutionary Algorithms are suited for parallel and distributed computing, a motivation for their popularity and some variants exist underlines this important aspect, e.g. the so-called Island Model where two or more populations evolve in parallel exchange solutions (migration) with a specified strategy. GAs may be adapted to multi-objective optimization problems and the most common is the category of Pareto-based approaches, using the concept of Pareto dominance, where the NSGA-II algorithm [57] is probably the most used. Evolutionary Programming (EP) and Evolution Strategies (ES), although presented earlier than GAs are scarcely used in the papers reviewed in this section. The first one, proposed by Fogel in 1962 [77], emphasizes the behaviour of the solutions (phenotypic evolution) and recombination operators are not present. Evolution Strategies was developed by Rechenberg in 1965 [192] and its first implementation, the (1+1)-ES, didn't make use of population. There exists for ESs some successful parameters adaptation strategies and nowadays this typology of EA is gaining more popularity thanks to a particular efficient implementation called Covariance Matrix Adaptation ES

Table 2.3: Evolutionary Computation approaches.

Approach	References with additional information
Genetic Algorithms	[144, 58, 108, 48, 221, 93, 34, 45, 35, 157, 239, 185, 158, 41, 38, 147, 222, 39, 231, 244, 177, 142, 80, 49, 240, 172, 178, 186, 152]
Island GA	[184, 128]
Multi-Objective GA	[230, 224] (MOGA), [176, 160] (NSGA-II), [127] (MOO)
Evolutionary Programming	[79, 78]
Evolution Strategies	[40], [126] (CMA-ES with Hybrid Differential Evolution)

(CMA-ES) [100].

All the EC-based approaches reviewed in this chapter are summarized in Table 2.3.

2.2.3 Other Computational Intelligence Approaches

Although neural networks and evolutionary computation cover the majority of the approaches reviewed in this chapter, other computational intelligence approaches are worth a mention. Particle Swarm Optimization (PSO) in one the most common Swarm Intelligence algorithms. Invented by Kennedy and Eberhart in 1995 [130], this algorithm is inspired by flocks of birds behave and decide their trajectories. In this computational methods, each solution (particle) moves into the solution space with a speed and a direction, influenced by both the solutions already explored and the best solution discovered from the entire swarm. For a detailed explanation of this algorithm see the book by Eberhart et al. [64] and the extensive survey by Poli [190]. Genetic Programming (GP) may be considered a specialization of a Genetic Algorithm where the solutions are tree structures expressing computer programs or, more in general, expressions with operators and terminals. John R. Koza is considered the ‘father’ of this methodology [137] and it has gaining a lot of interest thanks to the growth of CPU power in many research areas

2.3 Building Energy System

Energy is used in buildings for operating systems such as HVAC systems (heating, ventilation and air-conditioning), lighting, elevators, which are essential for the comfort and safety of building’s occupants. We can see all the problems related to these systems at different levels: building level, considering whole building energy management, and single system level.

2.3.1 Design and control of Thermal System

Inside a building, a comfortable internal environment is achieved by the use of HVAC systems and it's not strange that mostly of the applications of computational intelligence techniques are about these systems. The control and operations optimization of HVAC systems is strongly influenced by environmental conditions, first of all internal and external temperature, and other external factors which often are hard to model analytically.

Furthermore, HVAC systems are big energy consumers, especially in commercial buildings where air-conditioning systems account for more than half the total electricity consumption. Therefore, their efficiency has a significant effect on the overall energy performance of these buildings.

HVAC optimization

Dickinson and Bradshaw in 1995 [58] presented one of the first application of Evolutionary Computation to HVAC systems. Optimization of system parameters and operation scheduling is performed with a standard GA which maps directly the system model by the means of a model description language. The authors underline in their work the accuracy of the proposed methodology and the issues related to computational times. Huang and Lam [108] presented the application of GA with the objective of obtaining the optimal parameters of a PI controller. A comparison with Ziegler-Nichols method is performed taking into account overshooting and settling time, showing that the GA-based method presents better performance than classical methods. A more sophisticated application on the optimization of HVAC systems performances with regard to power saving (i.e. operating costs) is performed with GA in Congradac and Kulic [49], controlling the air damper in order to minimize the costs and the concentration of CO_2 inside the building. The HVAC system is modelled with MATLAB/Simulink software [162] and a GA control with three different target levels of CO_2 concentration is compared with the case without its use (standard operation mode). In all the cases the use of Genetic Algorithm provides an evident energy saving and the results are validated with a detailed model of a business building using EnergyPlus software [52]. The simulation has been performed on a summer working day and the energy consumption of chillers is reduced by $11-21\%$ respect to the standard operating mode which leaves the air dampers always opened. Other approaches based on evolutionary computation for HVAC system control can be found in Fong et al. [79] and Kie and Theng [133]. The first work uses Evolutionary Programming (EP) to optimize chiller water supply temperature and air temperature of air handling unit of an HVAC system and the latter minimizes the total power consumption of the condenser power loop by the means of GA. Both works show an effective saving potential of consumed electricity (respectively of 7% and 35%). The work by Nassif et al. [176] discusses a multiobjective GA-based approach for the optimization of HVAC control strategy, considering both energy demand and thermal comfort, measuring the latter with the predicted percentage of dissatisfied (PPD) [73]. This method, based on NSGA-II algorithm, optimises the various operating setpoints and it is applied on an existing HVAC system with a final achievement of 16% of energy savings on summer months. A complex HVAC system is optimized in Lu et al. [158] using an ANFIS system and using a GA with the aim of minimizing the overall energy consumption. In the GA, solutions represent the number of chillers, water pumps and cooling coil fans, the airflow rates of supply, and the temperature of chilled water supply. Due to the multiple nonlinear constraints, a penalty function is added to the fitness function in order to penalize infeasible solutions. The proposed method using ANFIS and GA is compared, by software simulations, with several traditional methods and it achieves, in all the cases considered, lower energy consumptions. In Ben-Nakhi and Mahmoud [20] GRNNs are used to optimize the scheduling of office buildings air conditioning simulating two buildings with ESP-r software [46]. The neural controller scheme proposed, composed by six different neural networks using as inputs hourly temperature readings, is designed to predict the time of the end of thermostat regulation aiming to have the setpoint temperature restored inside the building for the beginning of the working days.

In Morel et al. [171] a NN-based heating adaptive controller, called NEUROBAT, is developed and furthermore tested both by simulations and real building, with the objective of reducing consumptions, increasing comfort and minimising the maintenance. Neural networks are used to predict solar radiation and ambient temperature and to model the building behaviour, considering a 6 hours prediction time horizon. The comparison via software simulations has been performed considering three different commercial controllers, while for the real building the experimentations are performed simultaneously on two independent rooms: one controlled with a conventional controller and the other one with the proposed controller. In both cases NEUROBAT controller leads to better comfort and lower heating consumptions, $11 - 13\%$ reduction on a year. A NN controller is adopted in Liang and Du [154] to control the indoor thermal comfort level, measured it with PMV index (Predicted Mean Vote) [73]. The neural network takes two inputs: error between PMV set value and measured value and error derivative. The output of the network is the control of the HVAC system. The authors compare the NN controller with a PI controller, simulating setpoint variations and cooling load disturbances. The proposed NN controller exhibits a very smooth signal and better performances, moreover allowing to obtain high comfort levels and energy savings. A thermal comfort control model incorporating NN predictive model is presented in Moon and Kim [170] and experimentations were performed on a residential building. The model takes into account air temperature, humidity and PMV with the aim of reducing overshoots and undershoots of traditional control strategies normally due to the late thermal response of the building and lag-times of cooling/heating systems. The proposed framework, tested with real experimentations and software simulations, reduces overshoots and undershoots with higher thermal comfort and in some cases reducing the building energy consumption.

Zhou and Haghighat describe in their work [240, 241] a methodology to design a ventilation system for office buildings based on NNs and GA. In order to estimate the efficiency of each designed solution, CFD (Computational Fluid Dynamics) simulations are used and due to their high computational complexity (20 hours for each evaluation) a surrogate model, based on NN, has been used for the GA, which commonly requires an elevate number of fitness evaluations (in this case CFD simulations). The feed-forward NNs take as seven inputs the parameters of the ventilation system evaluated and estimate PMV, energetic consumptions, and other interesting variables. The training phase is performed on a set of initial simulations, selected with a Latin Hypercube Sampling (LHS) [113] method to sample the parameter space with the minimal number of points. Once trained the NN, this is used into the GA fitness function, which takes into account four objectives combined in a singe one with a weighted sum. Five different combinations of weights are considered and the improvement with respect to a baseline case is shown, leading to the conclusion that GA approach is critically dependent by weights values but potentially permits to obtain important improvements respect to classical designs. Ooka et al. [184] present an optimal design method for a building energy system, providing the best combination of equipment and operational planning with respect to consumptions and emissions. A distributed GA is used with its individuals' chromosome representing the capacity of selected equipments (cogeneration system, turbo refrigerator, heat pumps) for the optimisation of equipment case, otherwise, for operation control case, each chromosome represents hourly scheduling of load factor for each equipment. This approach is tested on a single day considering an hospital building and the comparison is performed with a exhaustive search approach leading to the conclusion that the multi-island GA algorithm used finds the most efficient solution.

HVAC control

With the aim of energy savings, a NN based controller which changes dynamically the room temperature according to thermal comfort value is developed by Yamada et al. [232]. The neural network with fuzzy control calculations controls the hot/cold water flow to maintain the desired room temperature in accordance with any deviation between computed PMV and target value. The network learns the correct thermal comfort model with the help of room users, which enter their thermal feelings through an input unit. The experimentation validated the approach which is able to achieve an energy savings of 18% respect to the conventional control strategy, with only the drawback of a PMV slightly greater than zero but however within the comfortable range.

Wang and Chen [223] developed a fault-tolerant control for buildings' ventilation based on neural network models. In order to keep an acceptable indoor air quality, CO_2-based demand-controlled ventilation (DCV) systems strongly depends on the information provided by sensors. The authors developed a system for detecting faults (classified as 'soft' and 'complete' faults) and validating measurements. The air conditioning system of an office building has been simulated with TRNSYS software [134] and data are collected to create NN models. Variables considered for the model are the damper control signals and air flows (outside and supply) and the fault-tolerant control strategy detects the anomalous deviations between models' estimations and sensor data to classify the fault typology and provide a correct measurement to the HVAC system. Validations tests were performed under various occupancy and weather conditions and it is shown than the proposed control strategy permits to achieve a good compromise between air quality and energy performances in case of sensor failures. A controller for building with high thermal inertia based on NNs is proposed by Argiriou et al. [10, 11]. It is composed by two modules: one for forecast energy requirements and a meteorological model for weather conditions (ambient temperature and solar irradiance). The meteorological part consists of two different feed-forward neural networks used as one-step ahead predictors, both have as inputs the past samples of temperature and solar irradiance and information about time and day of the year. Similarly, the heating energy predictor network has 35 inputs consisting of the previous samples of temperature (ambient and indoor), solar irradiance and status of the heating system. The neural controller developed has been tested online and offline: in the first case using a test room with a data acquisition system for three months, and in the other case via TRNSYS software simulation comparing it with a conventional controller. In both cases the proposed controller was able to maintain the temperature set point and the comparison with the conventional controller showed a saving of $7.5\% - 15\%$ on heating energy consumption. The paper by Mossolly et al. [172] examines optimal control strategies of air-conditioning systems applying a GA to the optimisation control strategies. With an objective function which takes into account comfort, air quality and energy consumptions, the GA optimises operational set points. The approach is validated on a case study of an academic building using Visual DOE software [3], with the conclusion that the optimised strategies lead to a consistent decrease of energy consumptions (from 10% to 30%) on summer period maintaining thermal comfort and indoor air quality.

Chiller Units

Optimization of chiller units capacity can lead to high energy savings, especially where the air-conditioning requirement lead to an high power consumption (e.g. industrial plants in hot areas). In Chang [41, 38] GAs are used to optimize the partial loading ratio (PLR) of chiller units, minimizing the energy consumption and satisfying the loading constraints. Evolutionary approach is compared on two case studies with Lagrangian method, which doesn't converge when the load request is below the 50% due to the non-convexity of the kW-PLR curve. The GA permits to reach satisfying results with an high execution speed. The same author in [40] use Evolution Strategies for the same task leading to similar results but simpler implementation. Another approach is proposed by Lee and Lin [150] based on Particle Swarm Optimization (PSO) algorithm. The authors compared this swarm intelligence approach with Genetic Algorithm and Lagrangian method on two test cases and in both PSO leads to lower consumptions on low demands.

Neural networks are used in Chow et al. [45] to model a direct-fire absorption chiller system and to be used by GA for optimization. Various settings, as pump speed or water temperatures, are used as inputs in a neural network with other uncontrolled variables (e.g. ambient temperature) and correlated with system's costs and efficiency. Once an effective model is created, GA is used to find the optimal set of control variables minimizing the costs. Comparison with standard designs on three cases shows the effectiveness of such approach achieving energy saving.

HVAC models optimisation

Advanced HVAC controllers integrate online predictors of system performance. To achieve an effective prediction, accurate models of measures like cooling and heating coil are needed to calculate optimal setpoints. Wang and Jin [221] use a GA algorithm for the on-line optimisation of HVAC control strategy parameters. A simplified physical model is used to predict the responses to the control set-points and the GA determines the optimal variables minimizing the cost function, which concerns PMV, air quality and energy use. As expected, the choice of the cost function shows to be critical and the experimentations, performed with TRNSYS software, demonstrate the effectiveness of control strategies proposed to improve overall system performances. Similarly, self-tuning models are developed in Nassif et al. [177] validating them on collected real data from an existing HVAC system. Model parameters are tuned by using a GA and the models include: zone temperature, return air entalphy/humidity, CO_2 concentration, fans, cooling and heating coil. The proposed model is compared with a simple one (without self-tuning) on measured data showing a significant improvement on accuracy.

Thermal comfort modelization

Normally human thermal comfort index is calculated using predicted mean vote (PMV) index [73] which may take a long computational time due to the nonlinear equations involved into the calculations. In Atthajariyakul and Leephakpreeda [12] NNs are used to learn the relationship between thermal variables and thermal comfort index, experimenting both models in an air-conditioned room and showing a good agreement between the two models.

2.3.2 Building Model Identification

The task of identification of characteristic parameters and coefficients of materials is very important to achieve an accurate modelling of building behaviour.

A procedure based on GA were proposed by Zhang et al. [239] to determining heat transfer coefficient of a wall surface. GA optimizes transfer function coefficients with as a goal the achievement of a particular value of heat flux error. Experimental data validates the proposed method and a high accuracy is achieved with a smaller computational load than traditional identification methods.

2.3.3 Intelligent Building and Ambient Intelligence

The term 'Intelligent building' refers to the advantages obtained by the application of information management and intelligent methods to the various building services and systems (see Flax [76]) with the aim to make them more efficient minimizing the costs.

Instead, the term 'Ambient Intelligence' gives a vision of a more pervasive presence of intelligence, dealing with the development of a new paradigm where people are immersed in a digital environment that is aware of their presence and context, and which is sensitive, adaptive and reactive to their desires, habits and emotions.

Lighting Controller

A self-adaptive management system which takes in account visual and thermal comfort inside the building and energy consumptions is proposed in Guillemin and Morel [93]. The method controls the blinds position according both to the exact position of the sun and the indoor lighting to optimize the visual comfort, e.g. avoiding too much solar light which can disturb the user. When the user is not present the controller set the priority to optimize the energy consumption. The method uses a heating controller which takes into account weather conditions (using a neural network based predictor), user presence, and temperatures. All the different models used: illuminance ratio, artificial lighting, climate, thermal room are based on fuzzy logic and NNs, and they are adapted using GA during the night. The method was experimented on two rooms inside an office building and compared with traditional controllers with the result of large energy savings (up to 25%) and good visual comfort (measured with user surveys).

Computational Intelligence Controllers for Intelligent Building

In an intelligent building a set of heterogeneous devices are controlled with the aim of satisfy users preferences, using all the data provided by sensors. Fuzzy controller or neural network can be used both as main tools, the former due to its ability to handle imprecision and the similarity with the human reasoning, the latter, on the contrary, is a black-box approach and permits to find even complex relationships between available data. Probably one of the first work about this topic is found in Mozer [173] where an adaptive control of an home is proposed. The authors developed a system which, observing inhabitant behaviours, is able to program itself adapting to their needs. The optimal control follows two objectives: minimisation of discomfort for users and energy cost. This system was implemented in a residence equipped with sensors in each room (temperatures, light, sound level etc) and with the possibility to control water heaters, ceiling fans, the intensity of lights, gas furnace, and electric space heaters. The global controller, receiving data from indoor and outdoor sensors, learns the habits and preferences of users (e.g. about light intensity) trying to minimise its objectives. Despite the authors didn't provide experimental data, the approach can be surely considered the first one about the utilisation of computational intelligence methods for home automation.

In Lopez et al. [157] an evolutionary algorithms is chosen for the generation of fuzzy controllers, experimenting and validating the technique on the real data collected on a dormitory (called iDorm) where sensors and actuators were available for the control during the staying, lasted several days, of a student. The algorithm is called GA-P, and it is an hybrid between GA and Genetic Programming (GP), and the deviation between the value proposed by the fuzzy controller and the real one of the actuator is used as metric of performance. An analysis of controllers is performed and interesting relationships between variables can be observed, moreover a comparison with ANFIS is carried out resulting in no significant differences in accuracy with the remark that ANFIS normally creates an higher number of rules than GA-P, but on the other hand, the GA-P tends to be slower than ANFIS.

2.3.4 Building and energy system design

Buildings are designed with the aim of providing a comfortable internal environment for the occupants, and the building envelope, which mainly consists of walls, roofs, windows, doors and floors, allows heat to flow between the interior and exterior of a building and, hence, it plays a key role in regulating the indoor environment. Therefore, the thermal characteristics of a building envelope have significant influence on HVAC systems, affecting, both, equipment capacity and energy required for their operation. Due to implementation difficulties and high costs, most measures to optimize the thermal performance of building envelopes need to be incorporated at the design stage of buildings or during a major upgrading exercise. Although such improvement measures are relatively expensive, they normally result in lower heating and cooling loads and so downsizing of equipment and lower energy consumption making them generally financially viable when considered on a life-cycle basis.

Given the large number of parameters often involved in building design process, multi-objective optimisation approaches are generally preferred given their capacity to consider all the objective functions (economical, environmental, etc) in parallel, without requiring explicit priorities.

Efficient building design

Optimization methods can be used in the design process for search and optimization of optimal design solutions. In Caldas and Norford [34] GAs are used to choose the optimal sizing of windows to optimize lighting, cooling, and heating performances. The effectiveness of a solution is provided by DOE-2.1E thermal simulation software [92]. The proposed optimization method permits to cope with large problems where there is no way to calculate manually the optimal solution and moreover it provides multiple solutions offering valuable alternatives to the designer. A multi-objective approach is proposed by the same authors [35] for the optimization of building shape and materials considering as objectives heating, lighting and costs.

The work by Wright et al. [230] proposed an optimisation approach for building design with the aim of finding the optimal trade-off between operating cost and occupants thermal comfort. The application is restricted to HVAC systems: setpoints, coil width and height, etc. Three days of operation have been used to evaluate the performance of the solutions provided by the used multi-objective GA implementation. Several design constraints have been taken in account and two optimization criteria have been specified: operating cost of HVAC system and maximum thermal discomfort (measured as PPD). The proposed approach offers the possibility to investigates the various design solutions showing great potential in the understanding of the behaviour of buildings and design solutions.

A multi-objective GA is used in Wang et al. [224] to design a green building performing a life cycle analysis (LCA), with the aim of obtain assist the designer to design a building considering both environmental and economical criteria. Both discrete and continuous variables are used to define the building design: orientation, aspect and window-to-wall ratio, window and wall types, materials of wall and roof layers, and roof type. Two fitness functions are considered: life-cycle cost (LCC) and life-cycle environmental impact (LCEI), both calculated using a simulation program based on a tool developed by ASHRAE [189]. A Pareto-based multi-objective GA is used and validated on a case study consisting of the design of an office building located in Canada. The Pareto front provided permits to understand the trade-off relationship between considered criteria and so the authors suggest that a multi-objective approach can assist the designer in the evaluation of the large amount of parameters involved in green building design.

A method to optimize the geometric form of a building in order to maximize the solar irradiation is proposed in Kämpf and Robinson [127]. They used RADIANCE [146] ray tracing software to simulate the irradiation of the building with the aim of creating a fitness function used by a CMA-ES/HDE hybrid Evolutionary Algorithm developed by the authors [126]. Three different test cases were considered: a grid of 25 Manhattan-style buildings, orientation and tilt of the roof surfaces of a set of buildings, roof geometry modelled as 2D Fourier series. The results showed that the evolutionary approach proposed was able to find good and, in certain cases, non-intuitive solutions raising an interesting question about the possibility of use such approaches as source of inspiration for architects and engineers. With the same objective is the work by Znouda et al. [244] which optimises the building shape with a classic GA implementation. The design variables considered are: length of the facades, types of roofing and walls and windows glazing. The evaluation of the solutions is performed by the means of a software tool to estimate the building behaviour called CHEOPS [84]. The authors performed the experimentations on two different optimization problems: one minimising the energy consumptions, and another on economic performance. Optimal solutions provided by the two formulations are, as expected, quite different, leading to the conclusion that a multi-objective approach should be preferable due to its feature of providing not only an optimal solution, but a set exploring the trade-off between the two objective functions.

A NSGA-II algorithm is applied in Magnier and Haghighat [160] with the aim of the optimization of building design. Several variables of the HVAC (i.e. set points and airflow rates) of a building are coded into solutions and two objective functions are used: total energy consumption (cooling, heating and fans) and average absolute PMV. In order to reduce the number of the time-costly building simulations (performed with TRNSYS), a neural network is used to model the building behaviour, providing an acceptable approximation. This approach leads to the creation of a set of Pareto-optimal solutions representing the trade-off between the two objective functions, with a significant improvement, especially considering the average PMV, with respect to the manually designed solutions

Energy System Design

In Fong et al. [78] Evolutionary Programming (EP) algorithm is used to optimize the design of a solar water heating system of a residential building. A model of a 28-stories building is built in TRNSYS and the following variables are included in the optimization process: tilt angle and surface azimuth of solar collectors, storage capacity of hot water calorifier and mass flow rate of circulation pump. This approach leads to the design of a more efficient system and it provided useful information for the engineering design.

In large building energy systems, e.g. in large residential buildings or hospital, it's not simple to find the optimal combination of machinery to optimize the overall efficiency of the system. In Kayo and Ooka [128] the authors present an application of island GAs for the optimal design of the energy system of a building. On two test cases, an hotel and an hospital, the size and presence of different types of machinery (e.g. refrigerators or gas boilers) are represented inside the individuals' chromosomes and the proposed design is simulated in order to calculate the primary energy consumption. For both cases the evolutionary approach performed effectively and it provided useful information for the design process.

GAs are used in Kumar et al. [142] to design earth-to-air heat exchanger in a non-air conditioned residential building. The algorithms optimizes five variables (e.g radius and thermal conductivity) using as fitness function the calculation of cooling potential of the system with a building simulation software. The solution found is compared with a deterministic model and a neural network for the prediction of the exit temperature of air, finally showing the best accuracy.

2.3.5 Prediction and forecasting

The availability of accurate and updated information may be fundamental for building energy systems for several reasons: to optimize the system parameters related to external factors, to obtain an effective control, to achieve an efficient design, etc. All the information commonly managed by Building Management Systems (BMS) can be used to predict future situations in the way to minimize the error and maximize the precision. Furthermore, forecasting data may be useful to find hidden relationship between available data.

To achieve an accurate forecast, information about the process we are coping with are needed but especially in real-world cases when important variables are not available, black-box approaches can be used. This kind of approach doesn't require any knowledge of internal dynamics but it considers the whole system only in terms of its inputs and outputs, the most common black-box methods are neural networks, which are able to model a system observing inputs and outputs. Especially in cases where a large amount of data is available, neural networks are commonly used in energy applications.

A review of modeling techniques for energy consumption, focusing on residential sector, can be found in Swan and Ismet Ugursal [211].

Energy consumptions

A modelling work of residential energy consumption with neural networks is performed in Aydinalp et al. [14, 16, 15] considering as input the properties of the heating system and the building, information about appliances and people living in the building, weather and temperature (indoor and outdoor) information and socio-economic characteristics of the building (e.g. income, dwelling type, etc.). Models were validated in the Canadian residential sector achieving a good prediction performance higher than engineering models already used.

Adaptive NN models are evaluated in Yang et al. [233] for real-time building energy prediction. Two adaptive models are proposed: one with accumulative training where the NN is retrained with new collected data and one with a sliding window, where the network is trained with a constant amount of measurements (older samples are discarded). Principal Component Analysis (PCA) is applied to input data in order to find and remove redundant variables. Static and on-line models, with and without applying PCA on input variables, were tested on data provided by DOE 2.1E software [92] simulation of an office building and on measured data collected from a real environment. Static models perform far better than on-line models on simulated data but the situation changes on real data, where all the proposed models don't achieve good performances. However, the work underlines the effectiveness of a PCA-based feature selection methodology.

Fewer inputs are instead used in the NN model by Dombaycı [61], which predicts the hourly heating energy consumption considering as inputs the hour of the day, day of the week, the month and the consumption of the previous hour. The model is tested on the consumptions of a residential building using as training and testing sets four years data, using different numbers of neurons into the hidden layer in order to find the optimal value which finally achieves about a 20% of relative error (MAPE) on the testing set. The work presented by Mihalakakou et al. [168] considers a six years period of time of the hourly consumptions for a residential building. Using as inputs the air temperature and the solar radiation, a neural network performs a prediction on a 1-year testing period with good results ($R^2 > 0.94$). Similarly, Neto and Fiorelli [180] consider the daily consumption of an office building testing two NNs: one with the minimum and maximum temperature as inputs, another with more weather data (relative humidity, solar radiation, etc). The latter network achieves testing error below 10% both for working days and holidays.

In Ekici and Aksoy [66] the yearly heating energy consumption of buildings with three different form factors are predicted with a neural network which has as inputs building orientation, transparency ratio, and insulation thickness. Several combinations of these inputs are used to simulate buildings' consumptions used as NN training set. All the buildings are considered to be in the same geographic area and the results show an high accuracy (94-99%). Wong et al. [229] create a ANN model able to predict daily cooling, heating, and lighting consumptions of a building given environmental, building coefficients and day of the week. Using EnergyPlus software [52], simulation data are provided to train the neural network model on an office building. The experimentations show that NN models used predict effectively the consumptions, achieving a maximum relative error of 8%.

The work done by Gonzalez and Zamarreño [87] uses a special neural network consisting of a MLP model with a feedback structure, trained with an hybrid algorithm composed by a backpropagation method with a random search. The predictor used considers as inputs the forecasting of the ambient temperature value, the hour, the day of the week, and the load at the previous time step. The method is applied on two benchmarking dataset provided by ASHRAE[1] for a competition and the results are compared with the winners. Also related to ASHRAE competition, in Dodier and Henze [60] the energy use prediction is operated by a NN with the application of statistical analysis to reduce the number of inputs. The authors used a neural network for each variable to be predicted, and for each of them a method based on Wald statistical test is applied to decide whether an input variable is relevant or not. Time and occupancy data were found relevant for all variables but not environmental data, furthermore an analysis of autocovariance was used to choose the time lag between values of inputs proposing a complete study of input selection analysis using NNs.

In Li and Su [152] an hybrid GA-HANFIS (Hierarchical Adaptive Network-based Fuzzy Inference System) model is developed and applied on the prediction of air conditioning energy daily consumption of an hotel. The inputs of the model are seven: the dry-bulb temperature and three past samples, and other three past samples of the daily consumption (the output). The GA optimizes the structure and the parameters of the hierarchical ANFIS and the authors try different combinations of inputs comparing the prediction accuracy of their

[1]American Society of Heating, Refrigerating, and Air-Conditioning Engineers

algorithm with a neural network, concluding that the former performs slightly better than the latter.

Steam load

Neural network ensemble is used in Kusiak et al. [143] with the aim of predicting the steam consumption. An initial feature selection is performed among the input variables and MLP neural networks are compared with other data mining method in finding relationships between the steam load and the weather data. The ensemble model outperformed the other methods and the prediction was effective especially during the heating season.

Cooling Load

A GRNN is used in Ben-Nakhi and Mahmoud [21] to model the hourly cooling load of a building with the aim of an optimisation of the thermal energy storage. Hourly ambient temperatures are used as input and a simulation software (ESP-R software [46]) is used to create the database needed for neural networks training and testing. The utilization of an ANN permits to use less weather inputs than building software simulations and thus predicting the hourly load, with effective results on testing buildings, using only simple data as external temperature. A comparative study of four modelling techniques has been presented in Li et al. [153], where support vector machines (SVM) and three types of neural networks are experimented on the hourly cooling load prediction of an office building. All the models use as inputs relative humidity, temperature and solar radiation, and the cooling loads used as values to compare are calculated with DeST software [234]. The experimentations show that SVM and Generalized Regression Neural Network perform better than a back-propagation neural network and a radial-basis function neural network.

Natural lighting levels

GA is used in Coley and Crabb [48] with the aim of predicting the natural lighting within a room, using the information of lighting sensors outside the building. The evolutionary computation approach is compared with a least-squares method and the former performs better with lower errors. Authors suggested it as the basis of a natural lighting controller and developed a prototype used to control the illumination within an office space.

Air flow

In Ayata et al. [13] NNs are used to predict the maximum air velocity and temperature into a building. Accurate simulations with the computational fluid dynamics software FLUENT [114] were performed and the simulation data are used to train an ANN using as inputs building parameters and wind information. The modelling work suggested a building dimensional ratio for an optimal choice of natural ventilation.

Indoor temperature and humidity

The issue of predicting building indoor temperature with the aim of control the heating system is addressed in Gouda et al. [91]. Four variables were taken into account: past history of outdoor and indoor temperature, solar irradiance, and heating valve position. A method based on Single Value Decomposition (SVD) is used to decrease the dimension of input data maximizing information content and the trained NN with this data is used to predict the indoor temperature up to two hours ahead.

Neural Network Nonlinear autoregressive (NNARX) models are used in Mechaqrane and Zouak [165] on the prediction of indoor temperature inside a residential building. First, an ARX model is applied with the appropriate number of past samples of considered variables (temperatures, solar radiation, and heating power) and then the same structure is used for the NNARX model. The neural network is optimized with a pruning procedure, the optimal brain surgeon (OBS) strategy [101]. The comparison shows that the NNARX model clearly outperforms ARX model, especially after the pruning procedure, which increased the prediction accuracy drastically. A similar approach is used in Lu and Viljanen [159] for prediction of indoor temperature and relative humidity. The variables considered are only outdoor and indoor temperature and outdoor and indoor relative humidity. A GA is used to determine the input variables and the number of hidden neurons of the network, coding into the chromosome all the possible combinations of inputs. As expected, the prediction of relative humidity is more difficult than temperature, and the GA performs better than NNARX methods on MSE measure. For indoor temperature from one-step to four-step prediction, both methods exhibit an high accuracy with the NNARX slightly better than GA.

For a similar task, a radial basis functions neural network approach is used by Ruano et al. [197]. All the work is based on environmental data collected from a secondary school building and the inputs considered are: air temperature (outdoor and indoor), solar radiation and relative humidity. The RBFNs can have as inputs one of the considered variables with a lag, for this reason there is a very high number of different inputs' combinations. The selection of pseudo-optimal inputs combination is performed by a multi-objective GA with 16 objectives: RMSE and mean error of training/testing/validation, amplitude of network's weights, number of inputs, and a set of correlation-based tests. The accuracy of optimal solutions is compared with the physical model showing that ANN perform generally better. Moreover, an adaptive version of the system is proposed in order to achieve high accuracy during the whole year using different observation sliding window sizes and then, both predictive models are used for the control of an air-conditioned system, demonstrating the possibility to achieve significant energy savings with the adaptive model.

Solar water heating system

A long-term prediction of solar energy output of a domestic heating system is performed in Kalogirou [122] using ANNs. The inputs used are: the month of the year, environmental parameters (temperature and solar irradiance), temperature of cold water, volume of the system, and various coefficients. The obtained percentage error is $2 - 5\%$ on different training sets.

CHAPTER 3

Ambient Temperature Modelling

"An expert is one who knows more and more about less and less until he knows absolutely everything about nothing."
Nicholas Butler

Organization of the Chapter

In this chapter, we introduce the application of neural networks to modelling, finally describing a specific application to temperature modelling.

1. Sections 3.1 and 3.2 describe neural networks and other algorithms later used in this chapter

2. Section 3.3 describes the ambient temperature modelling problem

3. An hybrid approach for neural networks training is presented in Section 3.4

4. Section 3.5 describes the specific problem and the experimental setup, where in Section 3.6 results of the application of various algorithms are examined.

3.1 Modelling with Neural Networks

Neural networks are powerful tools to solve complex modelling problems of non-linear systems. This technique gained a lot of popularity thanks to its advantages: easiness of implementation, the capability of be applied to an immense variety of problems where it performs reasonably well, and its general-purposeness. Neural networks demonstrated their effectiveness in system identification and modelling in many studies and real-world applications [9, 24, 175, 181, 31, 228, 50, 89, 32, 107, 198].

There are various architectures of neural networks, notable ones are feed-forward and recurrent networks. Feed-forward networks are commonly used in modelling and pattern recognition tasks while recurrent networks are used to construct a dynamic model of the process.

Neural networks are computational models which, observing outputs and inputs of a system, are able to reflect linear and non-linear relationship among them, given a specific error measure. This means that, having a non-linear function $y = f(x)$, we require that the neural network described by the function $F(\cdot)$ is close enough to $f(\cdot)$ as:

$$||F(x) - f(x)|| < \epsilon, \forall x \tag{3.1}$$

With enough training data and the right neural network structure we can have the approximation error ϵ small at will. Neural networks are considered a black-box methodology because they don't require prior information about the physics of the system. When a certain level of system knowledge is available (e.g. operating range, degree of nonlinearity, dynamics rapidity etc.) the modelling is called 'gray-box'.

This ability to approximate an unknown function, defined by its input-output mapping, is at the basis of any modelling task. Neural networks may be used to identify a system, learning the relationship between input (x) and output (y), and to perform the inverse task, treating x_i as the desired response to y_i, although this task can be very difficult because may not be a unique solution for a given output.

Modelling of any system with neural networks is performed in the following steps:

1. System inputs and outputs are selected
2. A dataset is created and, depending on the neural network typology and structure, it is pre-processed (e.g. scaling, filtering, normalization)
3. Dataset is split in two subsets: one used for the network training and the other one is used to test the generalization capability of the trained network (another subset may be used to decide the stop of the training algorithm)
4. Neural network is trained, i.e. weight coefficient values are determined respect of a specific error measure, usually Mean Square Error (MSE)

5. Generalization capability is tested on a data subset not used during the training phase, i.e. not observed by the neural network

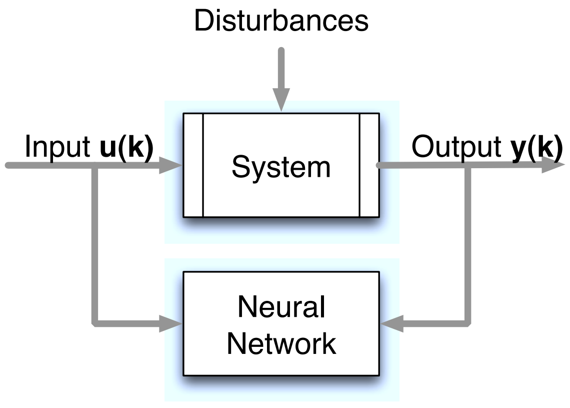


Figure 3.1: Modelling with a neural network

In modelling and regression problems, the common error measure is the mean square error (MSE). During the beginning of the training phase, the MSE is usually quite high. The expected behaviour is that as the network is trained, the error will gradually decrease until it reaches a minimum. It's worth noting that when only the MSE is considered as indicator of neural network performance, the network may tend to minimize the error giving as output always the mean value of the function. This means that the training methodology is a critical phase and problem-dependant, although many empirical rules exist about the kind of activation functions or training algorithm to choose (for an example see Lecun et al. [148]).

3.1.1 Structure of Neural Networks

Neural networks are structured with a set of interconnected layers, each of them composed of nodes; the typology of connections and nodes (called neurons) characterizes the different typologies of neural networks. In this work we considered the most common feed-forward (FF) neural networks: multilayer perceptrons (MLP) and radial basis functions (RBF) networks (see [102] for a detailed introduction). They are structurally equivalent and are considered feed-forward networks because the layers are connected starting from the inputs and arriving to the outputs with the information flowing in only one direction. The connections are weighted edges linking two neurons; weight's value is a real number normally initialized with a random value and it changes during the network's training. The nodes/neurons consist of a function, called activation function, generally a non-linear function which takes an input the sum of all the connected neurons' values of the previous layer (see figure 3.2).

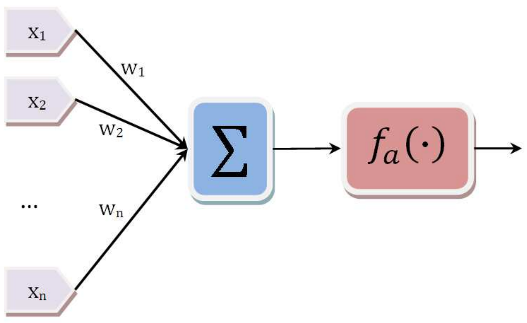


Figure 3.2: Neuron scheme

A complete introduction on neural networks and their theoretical background can be found in [102, 129].

3.1.2 Multi-Layer Perceptrons (MLP)

MLP neural networks can have one or more hidden layers, each of these is composed of non-linear activation functions, commonly a differentiable sigmoid function (generally a logistic function $\frac{1}{1+e^{-t}}$ or an hyperbolic tangent $\frac{2}{1+e^{-2t}} - 1$) which takes in input the inner product of inputs and connections' weights.

Learning process for MLP networks has to decide which features of the input pattern should be represented by the hidden neurons, which work as feature detectors, with respect to the error for each training pattern.

The back-propagation (BP) algorithm is the most commonly used technique for training neural networks, proposed first by Werbos [226] and later by Parker [188] and by Rumelhart and McClelland [199]. The algorithm is made up of two phases: a forward one, where the input is propagated through the network, and a backward phase, where the error signal is propagated from the outputs to the inputs calculating the adjustment for each layer. After the presentation of k-th input, each weight w_{ij} that connect the node i to the node j is corrected according the following formula:

$$\Delta w_{ij}(k) = \eta \delta_j(k) y_i(k) \tag{3.2}$$

where η is the *learning rate* parameter, y_i is the output of the neuron i, and $\delta_j(k)$ is the gradient defined as:

$$\delta_j(k) = \frac{\partial E(k)}{\partial v_j(k)} = e_j(k)\varphi_j'(v_j(k)) \tag{3.3}$$

where $e_j(k)$ is defined as the error between the output of the network and the desired value, $E(k)$ is the square error $\frac{1}{2}\sum e_j^2(k)$ for all the network outputs, v_j is the input value for the neuron j, and φ is the activation function.

As the back-propagation (BP) algorithm is an application of the gradient descent (also known as steepest descent) method to the weight space, it suffers of all the problems of this method (see [205]), for this reason the algorithm is common improved with the introduction of an adaptive learning-rate and a momentum constant, which make the algorithm more effective, stable, and with a smoother trajectory in weight space (see [102]). Many major modifications of BP algorithms have been proposed during the years, which have proved their usefulness in practical applications. It is worth mentioning the *quickprop* algorithm [70], resilient back-propagation [195], the Levenberg-Marquardt algorithm [98] and conjugate gradient methods [97].

3.1.3 Radial-Basis Functions (RBF) Networks

Radial basis function networks are neural networks whose hidden layer is composed of radial basis functions (e.g. Gaussian functions), functions whose output depends only on the distance of the input vector from a point defined *center*. Differently from MLP that can have more than one hidden layer, RBF networks have only one hidden layer. Hence, the output of the network is:

$$F(x) = \sum_{i=1}^{N} w_i^o \varphi(\|x - c_i\|) \tag{3.4}$$

with N the number of nodes in the hidden layer, w_i^o the weight of the connection to the output node, c_i the center of the function i, and $||\cdot||$ denotes usually Euclidean norm. If we define $\mathbf{\Phi}$ as the N-by-N matrix of all the elements

$\varphi_{ij} = \varphi(||x_i - c_j||)$, we may write:

$$\mathbf{\Phi w} = \mathbf{x} \tag{3.5}$$

and so we may find the weight vector $\mathbf{w}$ as:

$$\mathbf{w} = \mathbf{\Phi}^{-1}\mathbf{x} \tag{3.6}$$

assuming that $\mathbf{\Phi}$ is nonsingular (which is true if all the points in $\mathbf{x}$ are distinct).

Generally the radial basis function is a Gaussian function defined as:

$$\varphi(\|x - c_i\|) = \exp[-\beta\|x - c_i\|^2] \tag{3.7}$$

but other class of radial-basis functions can be used, the most common are the multiquadrics ($\varphi(x) = \sqrt{(x^2 + c^2)}$) and the inverse multiquadrics ($\varphi(x) = \frac{1}{\sqrt{x^2+c^2)}}$)

RBF neural networks' training can be separated in two problems: choosing the shape of the radial-basis function and choosing the output layer weights. A well-established training algorithm such as the back-propagation for MLP networks doesn't exist for RBF networks and different techniques were developed during the years.

A simple approach may be the following:

1. Placing the RBFs centers randomly or using a clustering algorithm (e.g. k-means)

2. Selecting the radius of the RBF (the β value for the common used gaussian functions in Eq. 3.7)

3. Computing optimal values of weights from the hidden to the output layer using a least-square method (or another optimization method).

For problems with small training data generally a RBF is placed in each input sample and the radius is chosen in order to ensure overlapping among functions, otherwise, for larger datasets, we can choose instead a random subset of input samples or use a clustering algorithm to determine RBFs coordinates.

It's worth noting that while MLP networks construct a global model, RBF networks, since they use exponentially decaying functions such as Gaussian, construct local approximations of non-linear relationships between input and output.

3.2 Other methodologies

There are a large variety of methods of interpolation and approximation, most of these methodologies can be found inside common technical computing software packages like MATLAB [162] or R [191].

3.2.1 Nearest Neighbour (NN)

This is a very simple method of interpolation, widely used to estimate unknown data providing as estimation the closest known point according to the following formula:

$$t = t_i \tag{3.8}$$

where t is the parameter to be estimated and t_i is the datum of the i^{th} closest point, therefore:

$$i = \min(d_j) \tag{3.9}$$

for $j = 1, \ldots, n$ where n is the number of known points and d a N-dimensional distance measure as the Euclidean Distance:

$$d_j = \sqrt{\sum_{i=1}^{N} (x_i - x_j)^2} \tag{3.10}$$

3.2.2 Support Vector Machines

Support Vector Machines (SVM) [220] are often associated to neural networks because they are universal approximators, able to perform classification and regression, but in many situations they outperform neural networks [33, 214]. SVMs perform a non-linear mapping on input vector into a high-dimensional space called feature space and construct an hyper-plane which separates the data following the structural risk minimization principle [220]. The application to regression problem is called Support Vector Regression (SVR) and was proposed in 1996 [63].

An interesting introduction to SVM and SVR could be found in [94, 179].

3.3 Ambient Temperature Modeling

The design of efficient solar based energy production systems and sustainable buildings strongly depends on simulations where the accuracy estimation of several environmental parameters is crucial. Among these, the most important ones are: solar radiation and ambient temperature. The first one is relatively simple to be modelled, because it mainly depends only on the geographical coordinates and the day of the year, and some work has already been done [117, 140, 206]. The second one is tougher because it depends on a high number of variables which are not always measured (e.g. wind). In this work we are facing the latter problem and we will show how, in the case of sustainable buildings design, an effective ambient temperature modelling tool can remarkably improve (about 20%) the energy consumption estimation error. Thus, when approaching this task two principal problems often rise:

- available data are based on monthly averages;
- existing data regard few places generally nearby airports

Concerning the first problem, algorithms which estimate reliable hourly values given the monthly ones already exist [69]. The second problem is slightly harder. So far, users of building simulation systems need to provide the tool with the most reliable meteorological data related to the location of interest. When these kind of data are not available for the specific required location it is common practice to use the data of the nearest unknown location (the Nearest Neighbour algorithm, NN). Unfortunately, this approach is not completely appropriate since climate is a highly non linear system and depends on a large number of variables. Indeed, locations geographically close to each other often have different environmental behaviours (e.g. example temperature profiles) and it is the cause of large errors.

Classical modelling approaches involve different interpolation techniques. Spatial interpolation makes it possible to estimate any meteorological characteristic (such as a maximum temperature) at locations away from those for which direct measurements exist. In this way, estimates can be made for scales up to continents and grid spatial resolution is typically in the order of several kilometres. The interpolation methods vary in complexity and accuracy, from simple Thiessen tessellation and inverse square distance [88] to more complex methods such as Truncated Gaussian Filter [119, 217], kriging and co-kriging methods and variations of spline interpolation [105, 116, 169]. The choice of methods is partly determined by the speed of computation required and nature of the modeled phenomena, whereas methods such as Thiessen polygon methods are very fast, kriging and multivariate splines require more computational effort. Applications range from [109], where meteorological stations were spatially interpolated over the whole of Europe using a multidimensional Regularized Spline with Tension (RST) [105] in order to get daily temperature profiles, to [187], where a procedure based on sets of equations to predict monthly mean values of relative humidity, ambient temperature and wind velocity for Indian locations is presented, to [110], where a method based on probability density functions to estimate daily temperature profiles is presented and applied to a few Australian localities.

Because of their capacity, neural networks are used for many different applications and tasks, works about the prediction and forecasting of indoor temperature and relative humidity can be found [132, 159, 19, 216] as well for the estimation and prediction of solar radiation [67, 145, 174] and ambient temperature [115, 86, 212, 75].

There are several applications of computational intelligence techniques in the field of modelling environmental parameters [6, 204, 207, 213, 8] which have provided interesting results.

In this context we are investigating neural network-based approaches in order to get more precise environmental estimation tools for ambient temperature.

3.4 Hybrid Training Algorithm for Neural Networks

As we stated before, the training task consists of the determination of the optimal weight coefficients respect to a specific error measure. Thus it may be formulated in an optimization form as follows:

$$\begin{cases} \min E(\mathbf{x}) = \sum_{k=1}^{N} (y(k) - \hat{y}(k))^2 \\ \hat{y}(k) = F(\mathbf{x}, \mathbf{u}(k)) \\ \mathbf{x} \in \mathbb{R}^{p \times q} \end{cases} \tag{3.11}$$

where y is the target data, $\hat{y}$ is the neural network output, $\mathbf{x}$ the vector of neural weights (and other parameters, e.g. bias), $\mathbf{u}$ is the input vector. Thus any optimization algorithm may be used to cope with this problem and given the complexity of the neural training problem in many real-world applications, we may use Evolutionary Computation algorithms.

The most common Evolutionary Algorithm is probably the Genetic Algorithm (GA), a class of algorithms inspired by natural evolution, proposed in the '70s [106], used for solving optimization and search problems in a wide domain. A GA operates starting from a population of solutions (represented with binary string or real-valued vectors) through a simple cycle of stages:

1. **Evaluation** of each string using a specified performance function (called *fitness* function)

2. **Selection** of the best strings

3. **Manipulation** to create a new population of solutions

This class of algorithms allows us to approach all kinds of problems, the only requirement is the presence of a performance function, which leads the “evolution” of the solutions toward the optimal.

3.4.1 Hybridizing Back-Propagation and Genetic Algorithm

Back-propagation (BP) and Genetic Algorithms (GA) previously introduced have been both use to train neural networks. Despite the success of such algorithms, each has its own merits and drawbacks.

The merits of BP are that the adjustment of weights is always towards the descending direction of the error function and that only some local information is needed. On the other hand, BP also has its disadvantages. For example, the error curve is generally so complex that there are a lot of local minima making the convergence of the algorithm very sensitive to the initial values.

GAs are parallel stochastic optimisation algorithms and compared to BP, they are more qualified for neural networks only whether the requirement of a global searching is considered. However, the price paid for GA is the slowness which is mainly due to the random initialisation of the genes and to the slow but crucial exploration mechanisms employed. Another shortcoming of GA is that the method cannot ensure convergence and achievement of the optimum. From this, it is easy to observe the complementarity between BP and GA.

GA and BP can be combined in different ways and two categories can be defined [201]: supportive, where one of these methodologies plays the primary role and the other one a supporting role, and collaborative, where they work together to solve the problem. The proposed hybrid approach, called BPGA, can be considered a supportive combination of BP and GA, indeed BP is first used to train several neural networks (a small fraction of the total GA’s population size) for approximately 10^6 cycles with no early stopping criterion. Then, the weights of the BP computations are encoded into several chromosomes of the GA’s initial population together with other randomly generated chromosomes (see figure 3.3).

The algorithm consists of two main stages:

1. Training of several ANNs with the back-propagation algorithm for a specified number of epochs, enough to reach the convergence

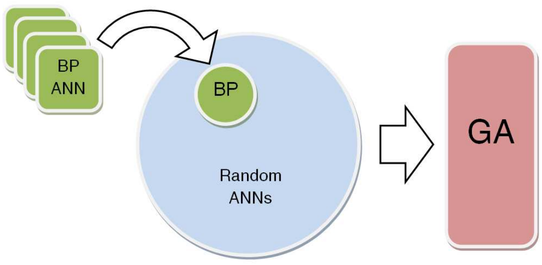


Figure 3.3: BPGA method

2. Using the GA to optimize the ANNs obtained at the end of the BP training, encoded in real-value vectors inside the individuals.

Therefore, in the proposed method the main advantage is that the searching domain of the GAs is reduced and thus, the convergence time is shortened. Moreover, the feature of parallel optimisation of GAs may help the BP networks to get out of the local minima which they tend to plunge into.

3.5 Experimental Setup for Ambient Temperature Modelling

In this work we compared six different methodologies on the modelling of temperature for Italian localities. The available data are the three geographical coordinates (latitude , longitude, height above sea level), the day of the year (1–365) and the monthly average temperature, therefore for each locality in the database there are twelve different temperature values, referring to the value of the middle day of the month. By a preliminary analysis we selected nine homogeneous areas in the Italian territory from the point of view of climatic characteristics (as reported in table 3.1), and so the whole data set, composed of 740 localities, has been split into nine subsets according to Italian defined climate areas. As required by NNs implementation procedure, each subset has been partitioned into two parts in order to proceed to the training and testing phases with different sets. In figure 3.4 the distribution of the considered localities on the territory is shown and table 3.1 shows the data set partitioning.

We considered all the following algorithms:

1. Nearest Neighbor (NN)

Table 3.1: Data set partitioning

	Regions	Training size	Testing size
Area1	*Valle d'Aosta, Piemonte, Lombardia*	76	17
Area2	*Trentino Alto Adige, Veneto, Friuli Venezia-Giulia, Emilia-Romagna*	161	18
Area3	*Liguria, Toscana, Umbria*	90	11
Area4	*Marche, Abruzzo*	43	13
Area5	*Lazio, Campania*	71	9
Area6	*Puglia, Molise*	66	8
Area7	*Basilicata, Calabria*	53	6
Area8	*Sardegna*	36	5
Area9	*Sicilia*	47	10
TOTAL	20	643	97

2. MLP Neural Network trained with back-propagation (BP)
3. MLP Neural Network trained with Genetic Algorithm (GA)
4. Support Vector Regression (SVR)
5. Radial Basis Function Networks (RBF)
6. MLP Neural Network trained with Genetic Algorithm with initial back-propagation solutions (BPGA)

MLP Neural Networks' architecture consists of 4 input neurons (latitude, longitude, height above the sea level and day of the year), 6 to 10 hidden neurons and one output neuron. Hidden and output neurons' transfer function is the logistic sigmoid function.

We implemented back-propagation and Genetic Algorithms in C++, for SVR we used the MATLAB interface for libSVM [37].

Algorithms' parameters are shown in table 4.6.

Tests have been carried out using all the data available for Italian localities in the database managed by ENEA (Italian Energy, New Technology and Environment Agency) [209].

3.6 Results

3.6.1 Monthly Temperatures

In table 3.3 and 3.4 we report the average and maximum absolute errors (with the standard deviations in brackets) averaged over 30 runs obtained on

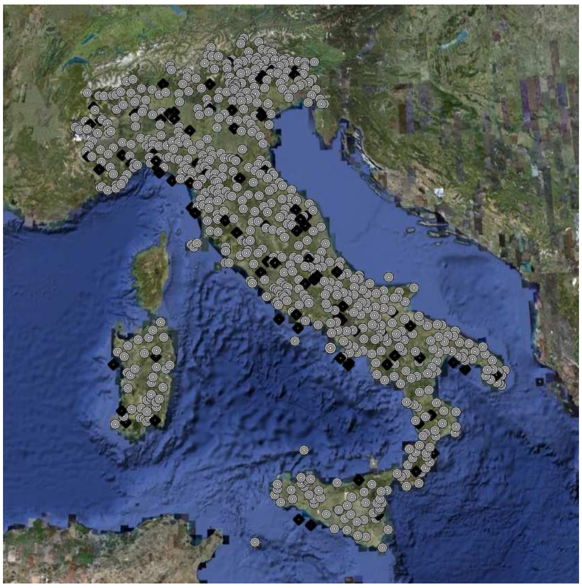


Figure 3.4: Subdivision of data in training (white circle) and testing (black diamond) datasets

the testing sets.

Experimentation clearly shows the effectiveness of the proposed BPGA approach. In fact, it outperforms all the other methods in terms of average and, more relevantly, it gives better performances than SVR even on maximum absolute estimation error. Furthermore, it is interesting to point out that the BPGA average standard deviation is very little (0.02° C), meaning that the method is robust and reliable compared to the other NN-based methods. To stress this achievement, in figure 3.5 we can see a graph representing table 3.3.

The reasons for this are mainly due to the fact that the searching domain of the GA is cut down by the BP initialisation and that the GA's parallel optimisation gets the BP out of the local minima in which it gets stuck.

Moreover, it is interesting to note that such results have been achieved using as input only the information available for every locality (i.e. geographical coordinates), without taking into account other important environmental parameters, like pressure, humidity and wind, which are accessible only for few localities (mainly those with an airport).

Table 3.2: Algorithms parameters

Algorithm	Parameters
Genetic Algorithm	Population size: 100; Crossover rate: 0.9; Mutation rate: 0.1; Stop criterion: 10^6 performance requests; Elitism: 1-elitism
Back-propagation	Learning-rate: 1.0
RBF Networks	MATLAB `newrb` algorithm; Maximum number of neurons: 50; Spread: 700
SVR	Typology: ϵ-SVR; Kernel: Radial Basis Function; ϵ value: 0.1; Cost: 400

Table 3.3: Validation results: average absolute error (°C). In **bold** is shown the minimum error for each area.

	NN	GA	BP	BPGA	RBF	SVR
Area1	1.07	1.16(±0.05)	0.80(±0.15)	**0.66**(±0.01)	1.01(±0.0)	0.71(±0.0)
Area2	1.47	1.12(±0.13)	0.86(±0.18)	**0.66**(±0.02)	0.97(±0.0)	0.68(±0.0)
Area3	0.93	0.81(±0.07)	1.12(±0.35)	0.70(±0.03)	0.89(±0.0)	**0.68**(±0.0)
Area4	1.0	1.98(±0.08)	0.79(±0.04)	**0.66**(±0.01)	1.29(±0.0)	0.74(±0.0)
Area5	1.59	0.77(±0.10)	0.60(±0.02)	**0.53**(±0.02)	0.90(±0.0)	0.57(±0.0)
Area6	1.28	0.85(±0.13)	0.72(±0.07)	**0.64**(±0.01)	0.80(±0.0)	0.70(±0.0)
Area7	1.28	1.06(±0.08)	0.75(±0.20)	**0.65**(±0.04)	1.33(±0.0)	0.78(±0.0)
Area8	0.65	0.84(±0.10)	0.99(±0.39)	**0.54**(±0.03)	0.81(±0.0)	0.64(±0.0)
Area9	3.11	1.88(±0.60)	1.10(±0.09)	**0.50**(±0.02)	0.96(±0.0)	0.82(±0.0)
Avg.	1.3	1.12(±0.15)	0.86(±0.17)	**0.62**(±0.02)	0.99(±0.0)	0.70(±0.0)

Finally, as an example we report a graph (figure 3.6) comparing the real monthly temperature to the one estimated by the nearest neighbour (NN) method and by the proposed approach (BPGA) over five localities belonging to Area 9.

3.6.2 Daily Temperature Estimation and Thermal Load Computation

As already stated in previous sections, one of the inputs of the proposed neural model is the day of the year (1–365) and the training data set has one "typical" day for each month (the middle day of the month) since data refer to monthly temperature. Therefore, such a kind of model can be used to provide

Table 3.4: Validation results: maximum absolute error (°C). In **bold** is shown the minimum error for each area.

	NN	GA	BP	BPGA	RBF	SVR
Area1	7.2	3.5(±0.20)	2.60(±0.60)	2.70(±0.02)	3.8(±0.0)	**1.97**(±0.0)
Area2	6.8	5.10(±0.30)	2.48(±0.65)	**2.40**(±0.10)	3.71(±0.0)	2.72(±0.0)
Area3	3.5	3.66(±0.10)	2.95(±0.80)	2.50(±0.05)	3.26(±0.0)	2.23(±0.0)
Area4	4.2	5.40(±0.15)	2.99(±0.07)	2.77(±0.02)	5.39(±0.0)	**2.21**(±0.0)
Area5	4.4	2.63(±0.12)	**1.47**(±0.13)	1.55(±0.05)	2.51(±0.0)	1.64(±0.0)
Area6	4.0	2.73(±0.11)	2.15(±0.40)	2.32(±0.02)	2.48(±0.0)	**1.97**(±0.0)
Area7	2.7	2.74(±0.25)	1.66(±0.36)	**1.65**(±0.15)	5.42(±0.0)	1.86(±0.0)
Area8	3.0	3.14(±0.10)	2.24(±0.34)	**1.75**(±0.05)	2.16(±0.0)	2.1(±0.0)
Area9	9.10	5.10(±1.50)	2.93(±0.47)	**1.77**(±0.02)	2.35(±0.0)	2.25(±0.0)
Avg.	4.99	3.78(±0.30)	2.39(±0.42)	**2.16**(±0.05)	3.78(±0.0)	2.2(±0.0)

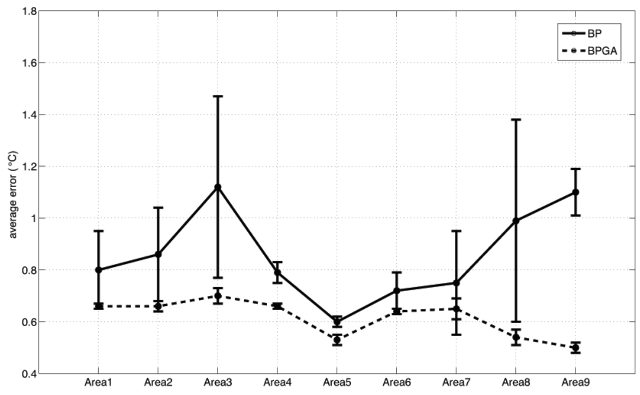


Figure 3.5: Comparison of average error and standard deviations of BP and BPGA.

daily temperature estimation over all the days of the year. This feature is very important when dealing with important design parameters like thermal load, where the daily ambient temperature estimation accuracy is critical.

Thus, we ran our models over all the 365 days of a year using as test case the city of Rome (for which real hourly temperature data are available) and then we provided the thermal load computation module with the neural-based

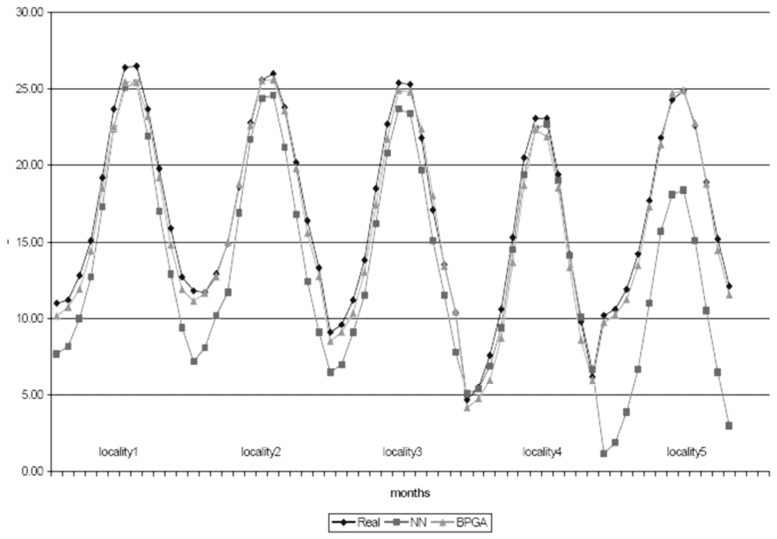


Figure 3.6: Comparison of the temperature modelling testing results (area9)

method outcome. In the thermal load computation we set the solar radiation to zero in order to see differences affected only by ambient temperature. The test was carried out on buildings with three different surface-volume (S/V) ratios (0.6, 0.5, 0.4) and three different windowing percentages (15%, 30%, 60%).

Thus, we had 9 different kinds of buildings (table 3.5) and we compared the different results of the thermal load calculation obtained by providing the simulation code (TRNSYS software [134]) with three different inputs:

1. hourly temperature from real data
2. hourly temperature estimate computed with the proposed approach
3. monthly average temperature (according to Italian regulation)

In the case A we provided hourly temperature data taken from meteorological stations (usually airports) and in case B hourly data estimated by our BPGA algorithm are considered. In case C temperature data are provided to software from a database which refers to data contained in Italian regulation, all the 101 Italian provinces. In the latter case commercial software we used

Table 3.5: Thermal load estimation test cases

	S/V	Windowing
BUILDING1	0.6	15%
BUILDING2	0.6	30%
BUILDING3	0.6	60%
BUILDING4	0.5	15%
BUILDING5	0.5	30%
BUILDING6	0.5	60%
BUILDING7	0.4	15%
BUILDING8	0.4	30%
BUILDING9	0.4	60%

Table 3.6: Results of the thermal load estimation (kWh) on the city of Rome with different setups: real temperature (A), daily temperature estimated with BPGA (B), and monthly temperature (C)

	Case A	Case B	Case C
BUILDING1	21 719	20 975	20 810
BUILDING2	28 012	27 086	26 841
BUILDING3	38 502	37 250	36 900
BUILDING4	37 315	36 020	35 745
BUILDING5	46 837	45 274	44 878
BUILDING6	64 251	62 135	61 572
BUILDING7	69 954	67 510	66 923
BUILDING8	88 530	85 526	84 739
BUILDING9	120 874	116 831	115 750
Average Thermal Load (kWh)	128 394	123 980	122 880
Average Absolute Error (kWh)		4 413	5 513
Absolute Percentage Error		3.4	4.25

applies a NN (see 3.2.1) approach whether the temperature of a location not in the database is requested.

From table 3.6 we can see that the daily estimation of the proposed neural model clearly outperforms the final thermal load computation reducing the error of about 20% compared to the monthly estimation.

We performed the same experiment we carried out on Rome on other 4 locations not present in the Italian regulation on which TRNSYS simulations are commonly based. At the time that we are writingwe had available real hourly data of year 2003.

Table 3.7: Thermal load and heating costs estimation on BUILDING2 for diffent localities. In brackets absolute percentage error is shown.

	Input Type	Casaccia	Piubego	Montalto	Portici
Thermal Load (kWh)	A	29205	34610	26951	13607
	B	29755 (1.9%)	35418 (2.3%)	27022 (0.3%)	25776 (89.4%)
	C	26274 (10%)	36928 (6.7%)	30217 (12.1%)	20858 (53.2%)
Heating Cost (€)	A	2684	3181	2477	1251
	B	2734 (1.9%)	3255 (2.3%)	2484 (0.3%)	2369 (89.4%)
	C	2415 (10%)	3394 (6.7%)	2777 (12.1%)	1917 (53.2%)

In table 3.7 thermal load and heating costs computed for the three different inputs are shown, in this case only one building typology has been considered. Costs are estimated considering a Lower Heat of Combustion of natural gas of $9.6\ \mathrm{kWh}/m^3$ with 0.85 of efficiency and a cost of 0.6 €$/m^3$ with a distribution efficiency of 0.8. As we stated before in the presented simulations solar irradiation has been set to zero, hence estimated heating costs should be higher than real ones.

3.7 CONCLUSIONS

In this chapter we tackled the issue of ambient temperature modelling since it is one of the most important environmental parameters when designing effective sustainable buildings. To solve this problem we proposed a hybrid approach based on computational intelligence techniques in order to provide ambient temperature for those places where such data are not available. Indeed, we combined the back-propagation algorithm and the simple Genetic Algorithm (BPGA) to effectively train neural networks in such a way that the BP algorithm initializes a few individuals of the GA's initial population.

Experiments concerned monthly estimation of unknown places and daily estimation for thermal load computation.

For the first problem, tests were performed over all the available Italian localities and results showed a remarkable improvement in accuracy compared to the single (BP and GA) and traditional methods (the Nearest Neighbour algorithm, NN). In particular, with respect to the NN approach (the most used in commercial software) the average modelling error is halved (from 1.3° C to 0.62° C) and the maximum is reduced by one third in the worst case (from 9° C to 2.8° C). Moreover, The BPGA method showed very high robustness and reliability (i.e. very low standard deviations).

In the second problem, we first compared the thermal load on 9 different building topologies in the city of Rome for which the software simulator we used (TRNSYS) has data. Then we focussed on one building and performed the same experiment on other locations for which the soft- ware simulator has no data and we compared the economic effect of the different approaches to the real situation by calculating the heating cost. This experimentation showed that the average estimation cost error is cut from 8.25% to 1.95% and it seems that the main benefit is when we have to deal with localities for which data are not known by the software simulator. This is a remarkable result because thermal load and cost consumption are the most important parameters in sustainable buildings and a bad estimation of these parameters might severely affect the design phase.

The reason for the success of the proposed approach is due to the fact that the BPGA algorithm combines BP and GA in such a way that the virtues of the single methods are enhanced. Indeed, the BP is first applied so that the searching domain of GA is trimmed down, reducing there- fore the GA convergence time, and then the parallel GA optimisation extricates the BP from the local minima which it plunges into.

Therefore, the main advantage of this method is that we have a non-linear interpolation tool capable of providing a reliable daily and monthly temperature estimate, which is critical for thermal load and heating cost estimations.

CHAPTER 4

Combined Cycle Power Plant Start-up Optimization

"New systems generate new problems."
Murphy's Technology Law

Organization of the Chapter

In this chapter we describe two different approaches of Evolutionary Computation to the optimization of the start-up phase of a Combined Cycle Power Plant.

1. A brief introduction of process optimization is presented in Section 4.1, focusing on the optimization of combined cycle plants

2. In Section 4.2 we present the specific application of Evolutionary Algorithms to the start-up phase optimization

3. Section 4.3 introduce the fuzzy modelization of the problem

4. The single-objective approach is described in Section 4.4 presenting the algorithms used and the results of the experimentations.

5. In Section 4.5 the multi-objective approach is analyzed with the results of its application on the real problem.

6. Section 4.6 concludes the chapter.

4.1 Process Optimization with Evolutionary Computation

With the class of Evolutionary Computation we refer to a population-based stochastic optimization process inspired by the principles of natural evolution. This approach allows to solve optimization problems where both the objective function and the constraints are black-box functions, without any information about Hessian or gradient. A large variety of algorithms and techniques go under the name of Evolutionary Computation but we can describe the generic procedure with the following simple equation:

$$\mathbf{x}^{t+1} = s(v(\mathbf{x}^t)) \tag{4.1}$$

where $\mathbf{x}$ is the population of solution at time t, v is a variation (mutation) operator, and s is the selection operator. There are a variety of solution representations and operators and the effectiveness of the algorithm depends on the choice of them, which is usually problem dependant. A complete overview can be found in [55, 65].

Evolutionary Algorithms can be applied virtually to any problem where an objective function (fitness function) can be defined. This function, which works as a performance index, is the first and the most critical choice when designing an evolutionary algorithm application. These algorithm allow to cope effectively with real-world problems, which normally include non-linear constraints, non-stationary conditions, noisy data, and other characteristics that makes them usually too hard for classical optimization techniques. The problem of find the best parameters for an industrial process, commonly referred with the name of 'process optimization', normally show all the characteristics listed previously.

In Figure 4.1 is shown the conceptual scheme of the process seen as a black-box. The process parameters $\mathbf{x}$ are the real object of the whole optimization, in fact the main objective is generally to minimize (or maximize) a cost function $f(\mathbf{x})$ with respect to inequality and equality constraints. An optimization problem involving multiple objective functions is known as a multiobjective optimization problem (MOP).

Since in MOPs objectives can be conflicting, such problems may lead to a set of solution instead of a single solution. Solutions belonging to this set are the result of a trade-off between conflicting objectives.

Given the complexity of real-world systems and industrial plants and the uncertainties involved in some design decisions, heuristic algorithms (and also Evolutionary Algorithms) are often applied in their design and improvement. Particularly, interactions among the various system components, the very large

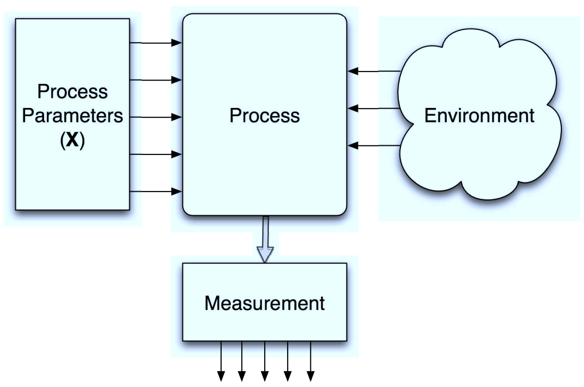


Figure 4.1: A process seen as a black-box

number of possible design alternatives, and the lack of accurate cost data for all plant components at an early stage of the design process make the optimization of real-world systems a difficult task.

4.1.1 Genetic Algorithms for Single and Multi-Objective Optimization

Genetic Algorithms (GAs) are probably the most known evolutionary algorithms, popularized by the work of John Holland [106]. Like others evolutionary algorithms its main application is for optimization purposes and it achieves its goal use a set (called population) of solution candidates to search the most promising areas of the solution space.

The simple (or canonical) GA follows the algorithm described in Algorithm 1 and originally it was designed with a binary string representation, proportional selection, single-point crossover and uniform mutation.

A real-value representation can be used instead of the binary one, which was considered at the beginning the best representation due to the Schema Theory, the original theoretical explanation of the GAs effectiveness proposed by Holland. Otherwise, a real-value representation gives several advantages: the discretization involved by the use of a binary string can be avoided thus, in case of continuous problems, representing more directly the variables object of optimization. Moreover, with a real-valued representation is simpler the definition of specific operators for a particular problem is simpler.

During the decades, a large amount of selection, mutation, and crossover operators and genetic algorithms variants have been presented. For an inter-

Algorithm 1 Simple Genetic Algorithm

```
1: $P^0 \leftarrow InitalizePopulation(N)$
2: $t \leftarrow 0$
3: while $!StopCondition$ do
4:     Evaluate($P^t$)
5:     $P^t_{\text{offspring}} \leftarrow \text{ApplyReproduction}(P^t)$ {Apply reproduction operators}
6:     $P^{t+1} \leftarrow Selection(P^t)$ {Select population for generation $t + 1$}
7:     $t \leftarrow t + 1$
8: end while
```

esting review see the book of Eiben and Smith [65] and the work by Reeves [193].

Multi-Objective GAs

Many real-world problems require the simultaneous optimization of two or more objective functions. Sometimes these objectives may be in conflict one with another, e.g. maximize the output of a power generator and minimize its emissions and consumptions. In multi-objective case instead of a single fitness function $f(\mathbf{x})$ we have a N-dimensional vector $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_N(\mathbf{x})]$. Obviously, in this case the definition of optimality is different from the traditional single-objective case.

The simplest approach is to ‘convert’ the multi-objective problem to a single-objective one aggregating the various objective in a single one, sometimes each one with a weight expressing its ‘importance’. Otherwise, we can use the Pareto theory with its concepts of Pareto optimality (or efficiency) and Pareto front, producing at the end of the optimization a set of non-dominated solutions (called Pareto-optimal set). A solutions $\mathbf{x}$ is defined non-dominated if doesn’t exist another solution which is better than it in at least one objective (see Figure 4.2 for an example).

The first implementation of Multi-objective Evolutionary Algorithms (MOEAs) was proposed by Schaffer in 1984 [202], called Vector Evaluated Genetic Algorithm (VEGA). A Weight Based Genetic Algorithm subsequently was proposed by Hajela and Lin [56]. The first algorithm that uses the non-dominated classification is the Multi-objective Genetic Algorithm (MOGA) proposed by Fonseca and Fleming in 1993 [81]. They proposed to assign a rank to each solution based on the number of solutions that dominates that one. This rank allows in some cases to compare two solutions without any fix-up like weights or other param-

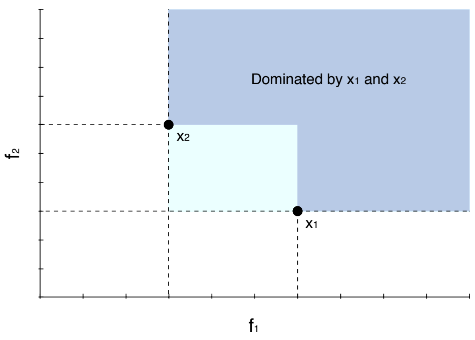


Figure 4.2: Illustration of Pareto dominance: solutions x_1 and x_2 are non-dominated and thus they make a Pareto-optimal set. A solution inside the grey area would be dominated by them.

eters. Subsequently many algorithms used non-dominated classification as the well-known Non-Dominated Sorting Genetic Algorithm (NSGA) proposed by Deb in 1994 [210] and then upgraded with elitism in 2000 with the name of Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II) [57, 56]. A survey about MOEAs can be found in [47].

To compare the quality of two solutions NSGA-II uses the ranking level approach. Given a population, this approach assigns rank level 1 to all non-dominated solution of the entire population, then it assigns rank 2 to all non-dominated solution of the population without solution of rank 1 and so on until all the solutions have been sorted.

In order to maintain a population as diverse as possible, a crowding distance value is assigned to each solution. This value describes how much is 'crowded' the solution space surrounding the solution, in fact it is high for isolated solutions and low for solutions with many neighbours of the same rank. Extreme solutions are always taken, with an infinity crowding value, and other solutions are compared to their nearest neighbours (see figure 4.3).

Algorithm 2 shows the pseudo-code of ranking level sorting of solutions in population. This algorithm is then used iteratively by NSGA-II during its optimisation process. Since Non-Dominated Sorting procedure is common and well-known in literature we omit to present its pseudocode. Subsequently the crowding distance value assignment algorithm is presented (Algorithm 3). Finally NSGA-II pseudocode is presented in Algorithm 4).

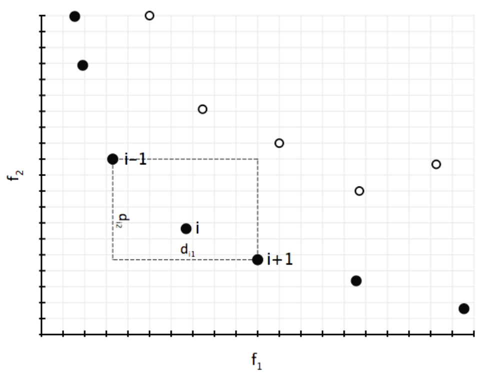


Figure 4.3: Crowding Distance

Algorithm 2 Ranking Level Sorting

```
Require: $P$ {Initial Population}
1: $j \leftarrow 1$
2: while $P \neq \emptyset$ do
3:     $P_i \leftarrow NonDominatedSorting(P)$ {Returns all non dominated individuals}
4:     $P \leftarrow P \backslash P_i$
5:     $j \leftarrow j + 1$
6: end while
7: $P \leftarrow P \cup P_i \forall\, i = 1, ..., j$
8: return P {Population ordered by rank level}
```

4.1.2 Application to Combined Cycle Power Plants

Combined cycle power plants (CCPP) are a combination of a gas turbine and a steam turbine generator for the production of electric power in a way that a gas turbine generator generates electricity and the waste heat is used to make steam to generate additional electricity via a steam turbine. The difference between this kind of power plant and a cogeneration plant is that the steam generated is used exclusively in the production of power, instead in cogeneration plants it can be used also for other purposes. As other industrial plants there is a large variety of optimization problems related to this typology of plants, some applications of computational intelligence can be found in

Algorithm 3 Crowding Distance Sorting

```
Require: $P$ {Population belonging to the same rank level}
Require: $M$ {Cardinality of objectives}
1: for $1 \leq m \leq M$ do
2:   AscendingOrderSort($P$,$f_m$) {Population sorted respect to $f_m$ values}
3:   $d_{(1,m)} \leftarrow d_{(|P|,m)} \leftarrow \infty$
4:   for $2 \leq j \leq |P| - 1$ do
5:     $d_{(j,m)} \leftarrow d_{(j,m)} + \frac{f_{(j+1,m)} - f_{(j-1,m)}}{f_m^{max} - f_m^{min}}$
6:   end for
7: end for
8: return $P$ {Population with crowding distance value assigned}
```

Algorithm 4 NSGA-II

```
Require: N {PopulationSize}
1: $P \leftarrow InitalizePopulation(N)$
2: $P \leftarrow RankingLevelSorting(P)$
3: for all rank levels do
4:   $P_i \leftarrow CrowdingDistanceSorting(P_i)$
5:   $P \leftarrow P \cup P_i$
6: end for
7: while $!StopCondition$ do
8:   $P_{selected} \leftarrow Selection(P)$ {Based on rank and crowding distance}
9:   $P_{offspring} \leftarrow Crossover(P_{selected})$
10:  $P_{mutated} \leftarrow Mutation(P_{offspring})$
11:  $P \leftarrow P \cup P_{mutated}$ {Applying Elitism}
12:  $P \leftarrow RankingLevelSorting(P)$
13:  for all rank levels do
14:    $P_i \leftarrow CrowdingDistanceSorting(P_i)$
15:    $P \leftarrow P \cup P_i$
16:  end for
17:  $ReplacePopulation(P, N)$ {Takes the best N individuals of P}
18: end while
19: return $P_1$ {Returns individuals belonging to the first rank}
```

[218, 136, 5].

For such plants, one of the most critical operations is the start-up stage because it requires the concurrent fulfilment of conflicting objectives (for instance, minimize pollutant emissions and maximize the produced energy). The problem of finding the best trade-off among conflicting objectives can be arranged like an optimisation problem. This class of problems can be solved in two ways: with a single-objective function managing the other objectives, like thermal stress, as constraints, and with a multi-objective approach.

Evolutionary algorithms have already been applied to the combined cycles power plants optimization. An application to the minimization of the production cost of complex combined cycle power plants is proposed in [136] where both the design configuration (process structure) and the process variables are optimized simultaneously. The work presented in [59] applies an evolutionary algorithm to optimize the feedwater preheating section in a steam power plant from a thermodynamic viewpoint. A power plant design problem is analyzed in [25] and the optimization, concerning techno-economic aspects, is carried out through multiobjective evolutionary algorithms.

At present, the problem of CCPP start-up optimization has been tackled in the first way using simulators. As example, in [7] through a parametric study, the start-up time is reduced while keeping the life-time consumption of critically stressed components under control. In [215] an optimum start up algorithm for CCPP, using a model predictive control algorithm, is proposed in order to cut down the start-up time keeping the thermal stress under the imposed limits. In [36] a study aimed at reducing the start-up time while keeping the life-time consumption of the more critically stressed components under control is presented.

In all the reported examples it is clear that the global start-up operations are not optimised. Therefore, in this chapter we propose an approach based on fuzzy sets in order to overcome the exposed drawbacks. Thus, for each single objective we define a fuzzy set and then we properly combine them in order to get a new objective function taking into account all the operational goals. We applied this method to a large artificial data set of different start-up conditions and we compared the best solution we found with the one given by the process experts.

In the last decade the application research of fuzzy set theory [243] has become one of the most important topics in industrial applications. In particular, in the field of industrial turbines for energy production, it has been mainly applied to fault diagnosis [182, 71], sensor fusion [85] and control. Particularly, in the last area in [26] it is proposed a fuzzy control system in order to minimize the steam turbine plant start-up time without violating maximum thermal stress limits. In [163] it is presented a start-up optimization control system which can minimize the start-up time of the plant through cooperative fuzzy reasoning and a neural network making good use of the operational margins on thermal stress and NO_x emissions.

EAs, as stochastic techniques, need an high number of evaluations of the fitness function to find the optimal solution and when the function is expensive (computationally or economically), as in real-world applications, it could be approximated to reduce the number of time-consuming calls, see [118] for a survey about this kind of approach.

4.2 THE COMBINED CYCLE POWER PLANT START-UP OPTIMIZATION PROBLEM

In this chapter we will use an Evolutionary algorithm to optimize the whole start-up process, this because EA will offer an easy and adaptable way to find an optimum in a complex function without the need of a deep knowledge of the process. This kind of algorithms are able to self-learn the trend of the objective function and seek for the best solutions in few steps compared with other optimisation algorithms.

Our main contribution is the application of computational intelligence methods to the global start-up optimization of such plants with a method for reducing the computational load of the optimization process.

In order to let the EA to work fine, we need to define a unique function that can represent the state of our process, considering a lot of variables (consumption, emissions, time, etc.) and merging them in a representative value. For this reason we have used a fuzzy set based fitness function which allows us to group many variables into a single value.

Gas and steam turbines are an established technology available in sizes ranging from several hundred kilowatts to over several hundred megawatts. Industrial turbines produce high quality heat that can be used for industrial or district heating steam requirements. Alternatively, this high temperature heat can be recovered to improve the efficiency of power generation or used to generate steam and drive a steam turbine in a combined-cycle plant. Therefore, industrial turbines can be used in a variety of configurations:

- Simple Cycle (SC): a single gas turbine producing power only
- Combined Heat and Power (CHP): a simple cycle gas turbine with a heat recovery heat exchanger which recovers the heat in the turbine exhaust

and converts it to useful thermal energy usually in the form of steam or hot water

- Combined Cycle (CC): high pressure steam is generated from recovered exhaust heat and used to create additional power using a steam turbine

The last combination produces electricity more efficiently than either gas or steam turbine alone because it performs a very good ratio of transformed electrical power per CO_2 emission. CC plants are characterized by high efficiency and possibility to adapt operation to different load conditions but they are an highly complex system which need the availability of powerful processors and advanced numerical solutions to develop high performance simulators for modelling purposes.

4.2.1 Start-up phase

The start-up scheduling diagram is shown in figure 4.4. From zero to time t_0 (about 1200 sec) the rotor engine velocity of the gas turbine is set to 3000 rpm. From time t_0 to t_1 the power load is set to 10 MW and then the machine keeps this regime up to time t_2. All this initial sequence is fixed. From time t_2 to t_3 (about 3600 sec) the machine must achieve a new power load, the initial set point load indicated as X_1, set point which has to be set optimal and then the machine has to keep this regime up to time t_4. The time lag t_4 ‚Äì t_3 is variable and is another variable to optimize, here called X_2, and during this interval the steam turbine starts with the rotor reaching the desired velocity. Then the turbines have to reach at time t_5 the normal power load regime (270 MW for the gas turbine) according to two load gradients which are variable depending on the machine; the gradient for both, compressor and steam rotors, are the last optimization variable taht we should use: X_3 and X_4. The sequence for that procedure is that first steam turbine grow up with X_4 gradient, then the turbine rotor can grow up following the X_3 gradient. In table 4.1 we report the process control variables (input) and the output variables to be monitored.

Therefore, the problem we are tackling has four inputs and five outputs and in order to optimise the overall start-up operations, the following objectives need fulfilling:

1. minimise time (Y1)
2. minimise fuel consumption (Y2)
3. maximise energy production (Y3)

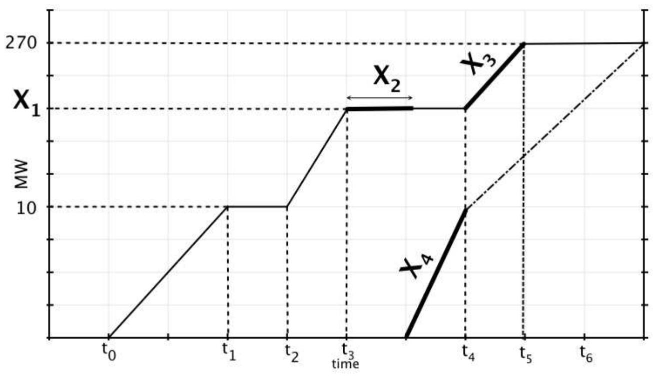


Figure 4.4: combined cycle power plant start-up operation

Table 4.1: Process input and output variables

Variable	Meaning	Operating range	Unit measure
	Input variables		
X1	Intermediate power load set point	$[20, 120]$	MW
X2	Intermediate waiting time	$[7500, 10000]$	sec
X3	Gas turbine load gradient	$[0.01, 0.2]$	MW/s
X4	Steam turbine load gradient	$[0.01, 0.2]$	%/s
	Output variables		
Y1	start-up time	$[11700, 29416]$	sec
Y2	fuel consumption	$[53000, 230330]$	Kg
Y3	energy production	$[6.45 \cdot 10^8, 4.56 \cdot 10^9]$	KJ
Y4	pollutant emissions	$[12.24, 32.58]$	$\frac{Mg \cdot sec}{Nm^3}$
Y5	thermal stress	$[8, 3939]$	-

4. minimise pollutant emissions (Y4)

5. minimise thermal stress (Y5)

In figure 4.10 a diagram with the correlation between each pair of objectives is shown, some linear relations are visually evident, e.g. between fuel consumption (Y2) and energy production (Y3).

4.3 Fuzzy sets definition

In order to allow a process of optimization through a black-box techniques such as Evolutionary Algorithms, we need to define a numerical quantity that can evaluate the whole process of start-up, giving an index of how the given configuration is effective, in harmony with the desired trend of the output values. The computed quantity will be used as a fitness value for our individuals in the evolutionary environment. In collaboration with process experts, we first defined the single fuzzy sets (see table 4.2) over the output variables (see table 4.1 and figure 4.5) and we composed them in order to get a cost function ranging in the range $[0, 1]$. Therefore, we got an index representing the global start-up performance. For every membership function, each linked to one of the process output, we used sigmoid membership functions with two parameters c and t:

$$\text{sigmoid}(x) = \frac{1}{1 + exp(\frac{c-x}{t})} \tag{4.2}$$

These functions are used simply as they are, if we wish to maximize the value, or used in a complementary mode, if we wish to minimize the output. The resulting fuzzy output has the following form:

$$\mu(y_1, y_2, y_3, y_4, y_5) = \sum_{i=1}^{5} w_i \mu_{F_i}(y_i) \tag{4.3}$$

Table 4.2: Fuzzy Sets

Fuzzy set	Membership function (μ_{F_i})	Variable	Weight (w_i)	t	c	Goal
F1	$1 -$ sigmoid	Y1	0.2	8000	110000	Min
F2	$1 -$ sigmoid	Y2	0.1	800	16200	Min
F3	sigmoid	Y3	0.1	$0.4 \cdot 10^9$	$1.8 \cdot 10^9$	Max
F4	$1 -$ sigmoid	Y4	0.3	2	25	Min
F5	$1 -$ sigmoid	Y5	0.3	20	150	Min

This composition has been finally chosen because we found out that for this problem the intersection was too restrictive (only one objective with a low value is sufficient to severely affect the whole performance) and the union was too lazy (only one objective with a high value is sufficient to have a high global

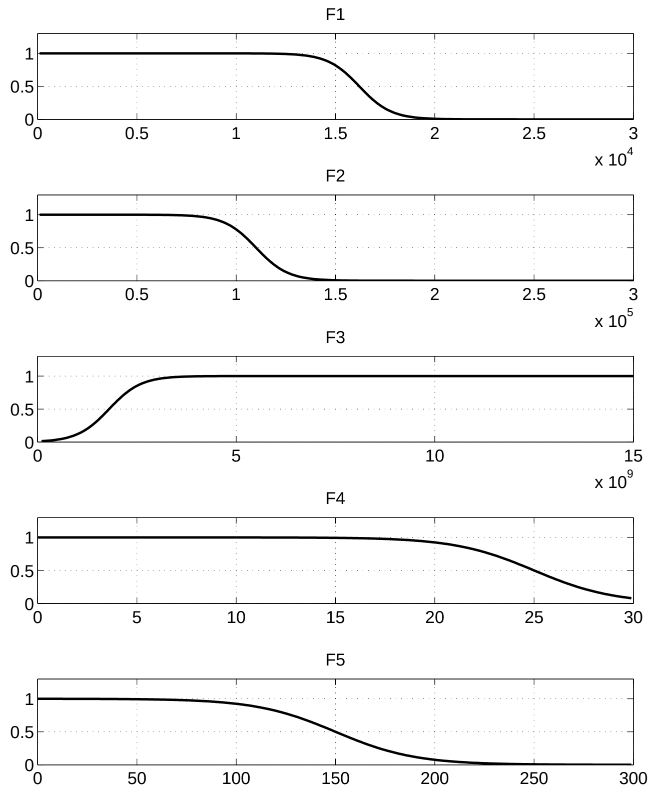


Figure 4.5: Fuzzy sets diagram

performance). Thus, we have finally applied the weighted sum operator, which is a good trade-off between intersection and union, which gives a global performance proportional to the optimality degree of each single objective. To obtain the weight for each fuzzy set in the previous composition we worked with the designer of this kind of Turbo Gas, in order to achieve a good combination of weights that can represent the theoretical directions that they try to reach when working on the start-up of this kind of process. With this function we try to work in cooperation with human behaviour, learning from the experience, instead of replacing the human factor.

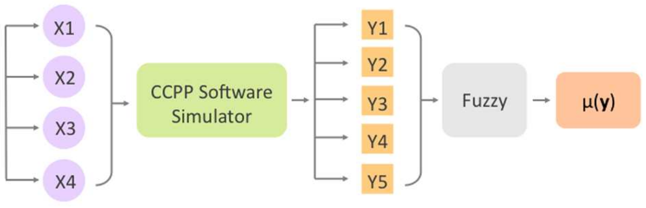


Figure 4.6: Diagram of the fitness model

4.4 Single-Objective Optimization with Evolutionary Computation

Evolutionary Computation methods have been used successfully in many optimization problems. The ability to perform a parallel search exploring in the solution space and exploiting the best solutions found is critical for the most complex problems. In our case the solution's genotype represents a start-up sequence encoding the variables described in table 4.1.

We implemented a real-coded Genetic Algorithm with a number vector's genotype representing normalized process' input variables. We choose a real-values encoding because of the continuous search space and in this way we avoided the discretization due to binary coding. The normalization of the input variables, between 0 and 1, is to make mutation operators' parameters heterogeneous given that inputs' variables differ strongly in magnitude (see table 4.1). A Gaussian mutation operator is implemented adding a random value following a normal distribution to the genotype's genes, i.e.:

$$g^i_m = g^i + \mathcal{N}(0, \sigma) \tag{4.4}$$

where g^i is the i-th gene and σ is the standard deviation of the gaussian distribution. We used a Uniform Crossover with a binary Tournament Selection and then as fitness function we use the fuzzy function shown in equation 4.3 (see figure 4.6 for a diagram of the fitness model), which is within the range $[0, 1]$. Two termination criteria have been set for this algorithm: maximum number

Table 4.3: GA parameters

Parameter	Value
Population Size	20
Mutation Rate	0.5
Mutation Amplitude (σ)	0.1
Crossover Rate	0.9
Tournament Pool Size	2
Max. number of generations	1000
Target fitness value	0.83

of generations and a target fitness value. Algorithm's parameters selected after a set of experimentations are shown in table 4.3.

4.4.1 Approximating the fitness function for computation load reduction

Algorithm 5 Calculate Approximate Fitness $f(x)$

```
Require: point $x$, archive $R$
 1: if distance$(x, R^x)$ < DISTANCE_THRESHOLD then
 2:     $j \leftarrow \text{nearest}(x, R^x)$ {Get the index of the nearest point from $x$ inside the archive}
 3:     $f(x) \leftarrow R_j^f$
 4: else
 5:     if distance$(x, R^x)$ < RANDOM_THRESHOLD then
 6:         $N \leftarrow \text{neighbourhood}(x)$ {Get the points which distance from $x$ is below the threshold}
 7:         $f(x) \leftarrow \sum_{i \in N} \frac{1}{1+\text{distance}(x, R_i^x)} R_i^f$
 8:     else
 9:         $f(x) \leftarrow \text{random}(0, 1)$
10:     end if
11: end if
```

Evolutionary algorithms applied to computational expensive problems, like the one considered in this paper, could be time consuming due to their stochastic nature. To tackle this issue we implemented an approximation method (pseudo-code is shown in algorithm 5) for the fitness with the purpose of reducing the number of fitness function calls.

All the points evaluated are stored into an archive R containing the point's n-dimensional coordinates and their fitness value in the last column, with the following form:

$$\mathbf{R} = [\mathbf{R}^x \quad \mathbf{R}^f] = \begin{bmatrix} x_{11} & \dots & x_{1n} & f_1 \\ x_{21} & \dots & x_{2n} & f_2 \\ \vdots & \vdots & \vdots & \vdots \\ x_{k1} & \dots & x_{kn} & f_k \end{bmatrix} \tag{4.5}$$

When the fitness value of a new point is requested, a search within the archive is performed to find a similar point, considering two points similar if their euclidean distance is below a certain threshold (DISTANCE_THRESHOLD), in this case we assume for the requested point the same fitness value of the similar one already inside the archive. Differently, if there is not a similar point, the method computes the fitness values in two ways: randomly, if the nearest point inside archive distance is above a threshold (RANDOM_THRESHOLD), otherwise interpolating the fitness value of the nearest points (see figure 4.7). The interpolated fitness of the requested point is obtained from a weighted sum of the nearest points' fitness values considering weights inversely proportional to the euclidean distance of the points (see line 7 in algorithm 5). At the end of each generation the best individual of the population is evaluated with the real fitness function and added to the archive.

The archive represents the information we have collected on the fitness model and the proposed method tries to approximate new points' fitness with an interpolation unless the point is too distant. In such case, randomness represents the lacks of information about that part of the fitness space and a random value enhances the possibility of explore unknown areas with the probability related to the fitness of the best individual. In fact, especially at the beginning of the evolution, a random value has an higher probability to have a better fitness value than the best solution already into the population.

4.4.2 Results

We performed 400 runs of the algorithm using the GA interfaced with the software simulator used to compute the fitness function value. In figure 4.8a is shown the distribution of the best solutions' fitness values at the end of the experimentations and in figure 4.8b the same for the number of generations.

The average number of generations is 414, i.e. the number of function calls is 8280 because at each generation a number of fitness evaluations equal to the population size is performed.

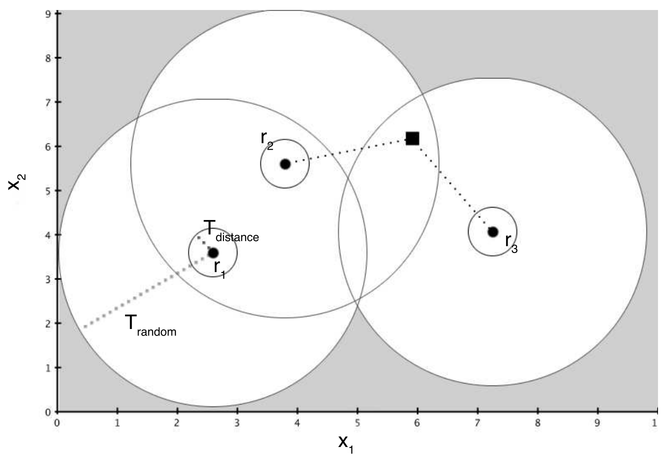


Figure 4.7: Example of the proposed approximation method: fitness value of requested point (square) is computed interpolating the fitness value of its neighbours (r_2 and r_3), i.e. all the points below the RANDOM_THRESHOLD (T_{random}) radius. The grey space is the part of the solution space where fitness value is computed randomly. $T_{distance}$ represents DISTANCE_THRESHOLD.

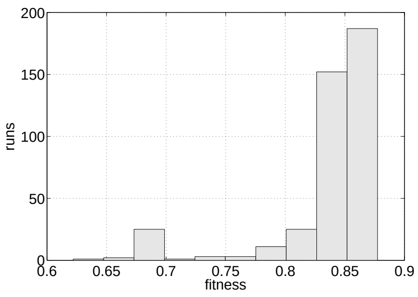
The same number of runs is performed with the fitness approximation method, in figure 4.8c and 4.8d are shown respectively the distribution of the best solutions' fitness values and the same for the number of generations at the algorithm's stop. In table 4.4 there is a comparison of the performance of both the approaches, with and without fitness approximation.

For DISTANCE_THRESHOLD and RANDOM_THRESHOLD we used respectively a value of 0.01 and 0.1, chosen after a set of preliminary tests.

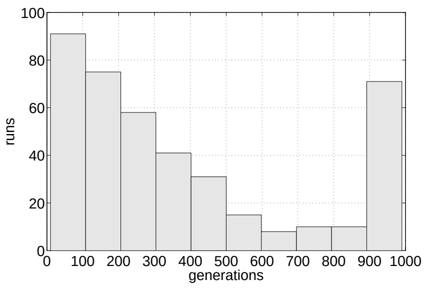
In the fitness approximation scheme we perform a single fitness function evaluation for each generation (the best solutions at the end of the generation), in this way an average run needs only 144 fitness function calls instead of the 8280 needed without fitness approximation.

We compared the optimal solutions found by both approaches with the solution provided by the experts, in table 4.5 we show the value of the five output variables (see table 4.1) for each solution and improvement of such solution calculated as:

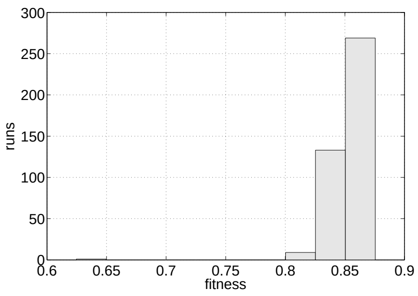
$$d_i = \frac{|Y_i - Y_i^e|}{\text{range}_i^{\max} - \text{range}_i^{\min}} \tag{4.6}$$



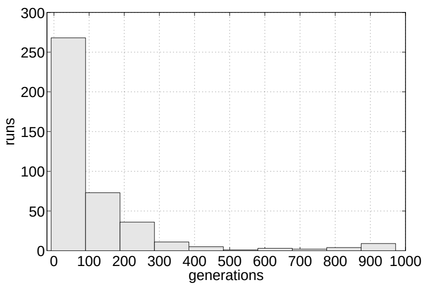
(a) GA: Distribution of fitness values.



(b) GA: Distribution of number of generations.



(c) GA with Fitness Approximation: Distribution of fitness values.



(d) GA with Fitness Approximation: Distribution of number of generations.

Figure 4.8: Results of experimentations

with Y_i^e is the i-th output variable of the solution provided by experts, $\mathrm{range}_i^{\max}$ and $\mathrm{range}_i^{\min}$ the operative ranges of the i-th variable (see table 4.1). The sign of the deviation is put positive if the deviation is considered an improvement, negative vice versa.

4.5 Multi-Objective Optimization with Evolutionary Computation

We applied our implementation of the NSGA-II algorithm on the multiobjective optimisation of the problem described in Section 4.1.1 and we compared it to the following algorithms:

1. RAND: A random search algorithm

Table 4.4: Experimentation Results

Genetic Algorithm (GA)	
Success Rate	79%
Average Number of Generations	414
Average Fitness Value	0.83
Average CPU Time per simulation	2070 hours
GA with Fitness Approximation	
Success Rate	98%
Average Number of Generations	144
Average Fitness Value	0.85
Average CPU Time per simulation	36 hours

Table 4.5: Comparison between solution provided by plants manager and best solutions of both approaches.

	Y1	Y2	Y3	Y4	Y5
Experts	21070	143557	$2.5 \cdot 10^9$	25	10
GA (values)	14800	99282	$1.5 \cdot 10^9$	21.6	54.3
GA with FA (values)	16569	115070	$1.86 \cdot 10^9$	18.8	78.4
GA (improvement)	35%	25%	-25%	17%	-1%
GA with FA (improvement)	25%	16%	-16%	30%	-2%

2. WSGA: Weighted-Sum Genetic Algorithm

3. NSGA-II: Non-Dominated Sorting Genetic Algorithm

The RAND algorithm is a trivial random search in the input space, in this case performing the same number of overall fitness evaluations of the other algorithms. At the end of this sampling, all the non-dominated solutions are considered inside the Pareto Front.

The WSGA applies a weighted sum of all objectives in order to reduce the original multi-objective problem to a single objective one. At each run a GA is executed with a different random convex combination of the weights of the fitness function.

All the algorithms were executed 10 times and the resulting non-dominated set of the union of the Pareto fronts obtained at the end of each run was taken.

In order to fairly compare the algorithms, each one is run over the same number of fitness evaluations.

Each input variable (see Table 4.1) can assume 21 different values, we encoded the decision variables with a Gray code binary string, whose minimal length can be obtained by:

$$\log_2 21^4 \approx 18 \tag{4.7}$$

The encoding is simple, we enumerated all the solutions (with numbers from 1 to 21^4) assigning each value of the 18-bit string to a solution.

Table 4.6 describes the algorithms parameters used during our tests.

Table 4.6: Algorithm Parameters

	NSGA-II	WSGA
Population Size	100	50
Generations	50	30
Selection	Binary Tournament	
Crossover	Single Point	
Crossover Probability	0.75	
Mutation	Bitwise	
Mutation Probability	1/18	

To evaluate the performance of the different methodologies we used the following metrics:

- Dominance Ratio
- Spacing
- Hypervolume

Dominance Ratio Suggested by Zitzler in 1999 [135], it compares two fronts and returns the fraction of solutions of the first one dominated by the second one, with respect to all the solutions of the first front. So given the fronts F^1 and F^2, the Dominance Ratio (DR) for the first front is defined as:

$$\mathrm{DR}(F^1, F^2) = \frac{\sum_{i=1}^{|F^1|} d_i}{|F^1|} \qquad d_i = \begin{cases} 1 & \text{if} \exists j | F_i^1 \succeq F_j^2 \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

With $|F^1|$ the size of the front F^1 and the symbol $\succeq$ indicating domination property, i.e. $i \succeq j$ means that solution i is dominated by solution j.

Therefore, a value of zero means that there are no solutions dominated by the other front and, otherwise, a value of one implies that the first front is completely dominated by the second. Since dominance operator is not symmetric, $\mathrm{DR}(\mathrm{F}^1, \mathrm{F}^2)$ is not always equal to $1 - \mathrm{DR}(\mathrm{F}^2, \mathrm{F}^1)$. It's clear that dominance ratio considers only dominance and doesn't describe totally the 'shape' of the two fronts.

Spacing This metric, proposed by Schott in 1995 [203], evaluates relative distance between consecutive solutions belonging to a set. A small spacing value describes a uniform the distribution of solutions within a front. Since in multi-objective problems is preferable to maintain the set of solutions as diverse as possible, as mentioned earlier, a uniform distribution of solutions is highly preferred.

Therefore given a front F of M objectives, spacing distance is given by:

$$S(F) = \sqrt{\frac{1}{|F|}\sum_{i=1}^{|F|}(d_i - \overline{d})^2} \tag{4.9}$$

where:

$$d_i = \min_{k \in F}\left\{k \neq i | \sum_{m=1}^{M} |f_m^i - f_m^k|\right\} \tag{4.10}$$

$$\overline{d} = \sum_{i=1}^{|F|} \frac{d_i}{|F|} \tag{4.11}$$

Hypervolume This metric was proposed by Zitzler and Thiele in 1999 [135]. It evaluates both dominance and spreading of solutions. This metric calculates the hypervolume whose vertices are the solutions set and a reference point, usually a vector of worst values each objective function can assume. Using the hypervolume of two fronts allow the comparison of both spread of solutions and their fitness on the various objectives. The calculation is computational expensive especially with an high number of objectives and an estimation based on Monte Carlo sampling[18] may be used. As reference point we considered the worst values among all the solutions of the Pareto fronts considered.

Thus, given a front F, if the real Pareto P^* front is known, it is recommended normalize hypervolume such that:

$$\mathrm{HV} = \frac{\mathrm{HV}(F)}{\mathrm{HV}(P^{*})} \tag{4.12}$$

Even if in real-world problems the real optimal Pareto front usually is not available, we computed, for a complete comparison of the selected algorithms, the fitness values of all the points inside the solution space. Despite it was computationally expensive (it took several days on a cluster with 1024 CPUs) we have the real optimal Pareto front

In order to show graphically the behaviour of the algorithms we tested the problem firstly for only two of the five objectives described in section 4.2. We considered two clearly conflicting objectives: maximization of energy production and minimization pollutant emissions. Subsequently we considered the problem with all five objectives and we present in this case only the related performance metrics results since the plot of Pareto fronts were not possible.

4.5.1 Results

We performed a multi-objective optimisation considering two objectives and five objectives. In the first case we considered the maximisation of energy production and minimisation of pollutant emissions and the overall number of fitness evaluations is 15300. In the second case we have the same number of fitness evaluations.

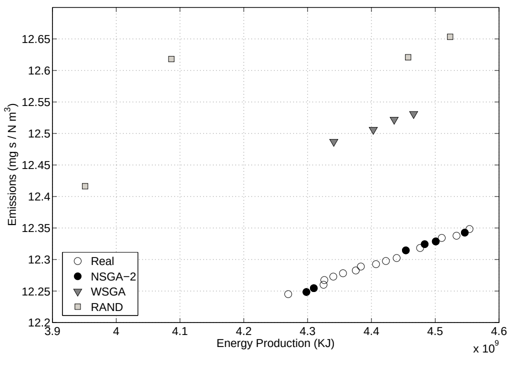


Figure 4.9: Pareto Fronts plot on the 2-objectives problem

Figure 4.9 shows that the NSGA-II Pareto front is overlapping with the real one and therefore it dominates all the solutions of the other algorithms while, as expected, the RAND front is dominated by both. The metrics values presented in Tables 4.7 and 4.8 reflect this situation. The columns labeled "2D" are related to the experimentations with 2 objectives and, similarly, for the problem with 5 objectives. The last line of Table 4.8 shows the number of solutions for different Pareto fronts.

Table 4.7: Dominance Ratio

	Real		NSGA-II		RAND		WSGA	
Dimensions	2D	5D	2D	5D	2D	5D	2D	5D
Real	-	-	0	0	0	0	0	0
NSGA-II	0	0.659	-	-	0	0.406	0	0.008
RAND	1	0.637	1	0.014	-	-	0.5	0.001
WSGA	1	0.5	1	0.1	0	0.1	-	-

Table 4.8: Spacing, hypervolume and size of the real optimal Pareto front and the ones obtained by the considered algorithms

	Real		NSGA-II		RAND		WSGA	
	2D	5D	2D	5D	2D	5D	2D	5D
Spacing	0.015	0.007	0.002	0.07	0.013	0.023	0.004	0.376
Hypervolume	0.93	0.394	0.898	0.348	0.069	0.37	0.338	0.129
Size	20	15608	11	261	4	2435	4	10

For the 5-objectives problem we can't plot directly the Pareto Fronts and so we have to establish the comparison between the algorithms on the metrics' values. We can observe that the size of fronts of the algorithms shows an evident variability: from 10 (WSGA) to 2435 (RAND) and the same we can assert the same for spacing, WSGA shows that the solutions in its front cover a larger space than other two algorithms.

4.5.2 Discussion

In two dimensions the results we obtain aren't much different from the ones we expected: the ability of Evolutionary Computation based algorithms like NSGA-II permits to explore effectively the solution space and find the best solutions, achieving a Pareto front far better than those obtained with WSGA or random search.

With 5 dimensions the situation changes drastically. The RAND algorithm becomes the best algorithm, achieving a Pareto front which dominates about the 40% of solutions of the NSGA-II's front and the nearest hypervolume to the optimal one. A probable explanation of this situation should be found in the last line of Table 4.8, where we can observe that the size of the real optimal Pareto front is about 800 times larger than the optimal one with two objectives. This means that random search is more effective because it's simpler to find randomly good solutions than in the 2D space.

It's an interesting observation the fact that the solution proposed from the plant manager results dominated in both the problem spaces, in 2 and 5 dimensions, by all the algorithms we tested. Therefore, all the solutions provided by the algorithms should be considered "better" (from a multiobjective point of view) than the real used ones.

4.6 Conclusions

When in an optimization problem the objectives are conflicting and subject to operational constraints, like in industrial applications, black-box approaches like Evolutionary Algorithms might give good performances due to their stochastic nature, assuming that an effective problem's representation could be found. We coped the start-up phase multi-objective optimization problems with two different methodologies: single-objective reduction with expert knowledge modelling and Pareto-based optimization, both applying evolutionary algorithms.

A major drawback for stochastic algorithms such EAs can be the high number of fitness evaluations needed in order to explore the solution space and find the optimal solutions. In applications where fitness function is particularly time-consuming, like the one in this paper, we tried, with the fitness approximation approach, to interpolate the fitness value of the new points from the solutions already evaluated assuming a static environment where the fitness value of a solutions doesn't change during the time. Despite the interpolation we implemented is not complex it provides better performances in the application of a genetic algorithm, leading to a reduction of the overall number of fitness function evaluations avoiding the evaluations of similar or identical solutions.

For the single-objective approach, we obtained in our tests a strong reduction of the number of fitness evaluations and a consequent decrease of the time needed for the optimization of the start-up phase from 2070 hours to 36 for 100 simulations. All the solutions found lead to a start-up sequence which is better than the already used one according to the plants operator and the results which show (see table 4.5) an improvement in three objectives and a worsening (in energy production).

We underlined the capability of multiobjective optimisation techniques of providing a set of feasible solutions among which a decision can be taken. We made our experimentations on a precise software simulator of a combined cycle plant considering two and five objectives functions.

Considering only a subset of the objectives (maximisation of energy output and minimisation of pollutant emissions) we observe that NSGA-II algorithm works far better than a random search and a combined single-objective algorithm, finding solutions on the real optimal Pareto front. With all the objectives the situation changes and the results of a random search outperform the Evolutionary Computation based approach. This fact may be explained by an intrinsic weakness of NSGA-II algorithm for particular problems, confirming the results presented in [131].

Despite these results seem inconsistent, we think that it is not simple to estimate the performances of a set of algorithms when increasing the number of considered objectives, because in real problems the objectives function to minimise (or maximise) are heterogeneous, i.e. the relation between results in low and high dimensional space is not straightforward. In the real case we considered, a deeper study of objective functions is needed, in order to explore mutual relations between them. However, an improved algorithm which has demonstrated its effectiveness for other multi-objective problems (e.g. HaD-MOEA [225]) might be applied in the future.

Although the primary goal of this paper is to highlight the application of multiobjective optimisation to a real world problem, comparisons can be extended also to other MOEA for a more complete overview.

Moreover this work raises the issue of reducing the computational load of stochastic algorithms such the ones we used of real problems, where the evaluation of a solution is based on the execution of a software simulator, which reflects the complexity of the problem it simulates. We think that such problem can be coped with by considering an algorithm which uses both the real fitness function and an approximated one, in order to lower the number of executions of the computationally expensive software simulator.

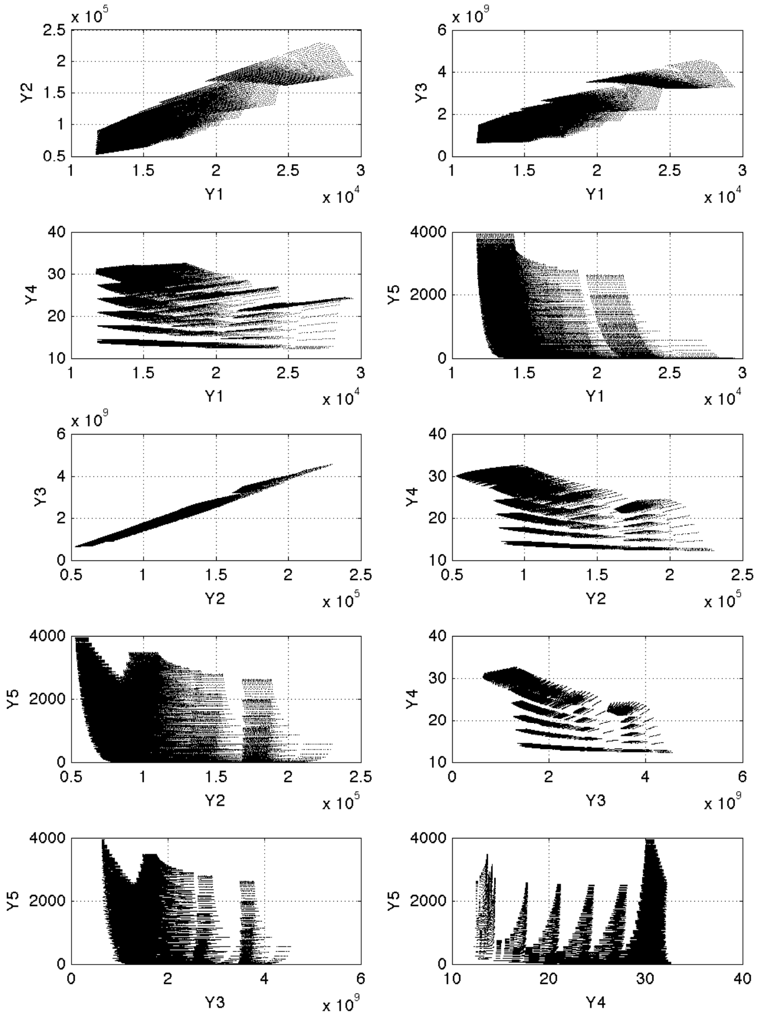


Figure 4.10: Plot of relations between objectives

CHAPTER 5

Load Forecasting with Neural Networks

"Rule of Accuracy: When working toward the solution of a problem, it always helps if you know the answer. Corollary: Provided, of course, that you know there is a problem."

Organization of this Chapter

This chapter is divided in 8 sections with the aim of describing a particular application of neural networks to short-term load forecasting.

1. Section 5.1 gives a brief introduction about the application of neural networks to the forecasting, describing the most common predictive structures. Moreover, neural ensembles are presented and for sake of completeness, traditional statistical models are outlined.
2. A brief introduction about short-term load forecasting is presented in Section 5.2.
3. Section 5.3 presents the specific application object of this chapter and consequently in Sections 5.4 and 5.5 used models are described.
4. Results of the real-world application are shown in Sections 5.6 and 5.7 and finally Section 5.8 gives some concluding remarks.

5.1 Forecasting with Neural Networks

Most time series forecasting methods use statistical approaches or artificial intelligence algorithms. The most applied methods are Box-Jenkins approaches, exponential and Holt-Winters methods and Neural Networks (NN) based methods. A general introduction to time series applications can be found in Brockwell and Davis [29] and a good survey on various methodologies applied to load forecasting is in Feinberg and Genethliou [74].

Theoretically, neural networks are able to model data as well as traditional statistical methods [227]. This ability, with their inherently nonlinearity, makes them a well-suited method for forecasting problems.

5.1.1 Neural Networks Structure for Forecasting

Neural networks have been applied successfully to a wide variety of forecasting problems. The main issue to solve the problem of forecasting with neural networks is the selection of the best structure to use. As we have seen in Chapter 3 there are several neural network structures, each one with its advantages and drawbacks for prediction tasks. The choice of the best structure is considered problem-dependant, for this reason there are in literature works suggesting empirical rules to build a neural network for a particular problem, see for example [44, 194].

Most widely used are feed-forward neural networks such as multilayer perceptrons (MLPs), and their weights training is usually performed with the well-known back-propagation algorithm.

The simplest neural network-based prediction structure we can consider for the forecasting is the one which takes in input the lagged samples of the output as:

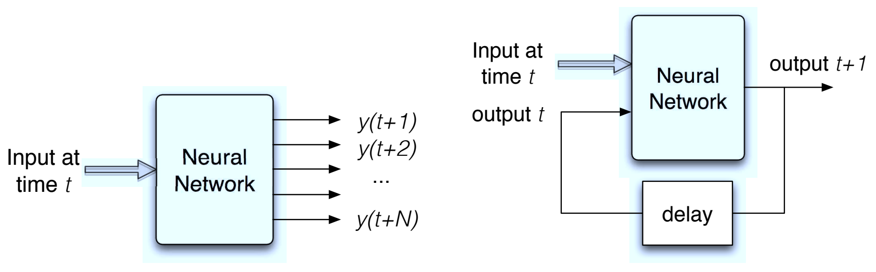
$$\hat{x}_{t+1} = f(x_t, \ldots, x_{t-N}) \tag{5.1}$$

The value N is the length of the data window considered and it constitutes, together with the choice of the lags, a critical design factor, particularly case depending. The structure described by Eq. 5.1 can be expanded to introduce additional inputs (I), obtaining the following one:

$$\hat{x}_{t+1} = f(x_t, \ldots, x_{t-N}, I_{t+1}, \ldots, I_{t-K}) \tag{5.2}$$

The number of inputs nodes is a critical variable for the forecasting application, in fact a low number may not provide enough information for an accurate forecasting and a too high number could make the training less effective, due to a larger and more complex solution space. The choice of the number of hidden layers used and the nodes included into each of them is usually made following some rules of thumb [120, 238].

Another factor that may affect performance of neural network for time series forecasting is the number of output neurons. If the we want to forecast at time $t + 1$ starting from time t (forecasting horizon of one period), then the number of output neurons is obviously one. When the forecasting horizon is greater than one, the number of output neurons varies according to the approach being used. The 'direct' forecast method puts the number of output neurons equals to the forecast horizon (see Figure 5.1a). On the other hand, if the iterative forecast method is adopted, the number of output neurons is equal to one. The predicted value is used as an input for the successive period prediction, until the end of the forecast horizon (see Figure 5.1b). This way of forecasting is the same approach that is used in Box-Jenkins models (see 5.4.2).



(a) Direct Method (b) Iterative Method

Figure 5.1: Neural network approaches for forecasting horizons greater than one

In order to measure the forecast accuracy there are many error measures that can be used, each of them trying to represent with a value the accuracy of a specific model on a dataset. The most simple is the Mean Squared Error (MSE), used normally for the back-propagation algorithm, but some more specific measures for forecasting problems are the following: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Coefficient of Variation of Root Mean Square Error (CV-RMSE), Mean Bias Error (MBE), and others. See the work by Hyndman and Koehler [112] for an insight about error measures.

Many competitions have been organized in an attempt to stimulate the development of neural network approaches in time series forecasting. The NN3 and NN5 competitions [2] focused on systematic approaches for predicting a large number of time series of a similar nature over forecasting horizons of 18 and 56 steps ahead, respectively. Also ESTSP conference (European Symposium on Time Series Prediction) and NISIS (Nature-inspired Smart Information

Systems) organized various competitions in the past, see for example [200, 155].

An interesting comparison between traditional statistical methods and neural networks can be found in Remus and O'Connor [194] and the work by Zhang et al. [238] gives an exhaustive review of the application of neural networks to forecasting application.

5.1.2 Neural Networks Ensembles

The term 'ensemble' describes a group of learning machines that work together on the same task, in the case of neural networks they are trained on the same data, run together and their outputs are combined as a single one [99].

Generally speaking, we can divide all the ensemble methods in three approaches: in the first one, all the predictors are created starting from a different subset of the training dataset, in the second one the various models work on the same dataset which samples are weighted following a particular strategy. In the last approach the models interact during the training phase and their outputs combined in some way.

Bagging [28] is a popular ensemble method that creates estimators training each of them with a random subset of training data. Instead, boosting methods [83] give more importance to the training samples which are predicted incorrectly more frequently than others. However, also a simple majority rule might be considered an ensemble method, as it is shown in the famous work by Hansen and Salamon [99].

An ensemble method different from the ones cited above is the Negative Correlation Learning (NCL) introduced by Liu and Yao [156]. This method adds a correlation penalty term to the error function used for the back-propagation algorithm in order to create some 'specialists' inside the ensemble.

Some examples on the application of neural network ensembles to forecasting can be found in [4, 237, 161, 17].

5.1.3 Other models

Commonly, a time series model (or Box-Jenkins model) is certainly considered the first choice in approaching a forecasting problem, especially the autoregressive integrated moving average model (ARIMA) which considers the nonstationarity of the data, presented in the landmark work of Box and Jenkins [27]. In this model future value of a signal is assumed to be a linear function of past observations with the addition of an error term (assumed with zero mean and indipendently and identically distributed).

Once introduced the backshift and the first difference operators as:

$$B^k x_t = x_{t-k} \tag{5.3}$$
$$\nabla^n x_t = (1-B)^n x_t \tag{5.4}$$
$$\nabla_s^N x_t = (1-B_s)^N x_t \tag{5.5}$$

a seasonal ARIMA model denoted as $\mathrm{ARIMA}(p,d,q) \times (P,D,Q)_s$ has the following form:

$$\Phi_P(B^s)\phi(B)\nabla_s^D \nabla^d x_t = \alpha + \Theta_Q(B^s)\theta(B)e_t \tag{5.6}$$

where x_t is the value of the signal at time t and e_t the error term (supposed to be a white noise process). The terms d and D represent the degree of differencing and the operators $\Phi_P(B^s)$ and $\Theta_Q(B^s)$ are respectively the seasonal autoregressive and the seasonal moving average operators of orders P and Q as:

$$\Phi_P(B^s) = 1 - \Phi_1 B^s - \Phi_2 B^{2s} - \ldots - \Phi_P B^{Ps} \tag{5.7}$$
$$\Theta_Q(B^s) = 1 - \Theta_1 B^s - \Theta_2 B^{2s} - \ldots - \Theta_Q B^{Qs} \tag{5.8}$$

The non-seasonal operators $\phi(B)$ and $\theta(B)$ are similar to the seasonal ones (eqs. 5.7 and 5.8) but considering the seasonality $s = 1$.

The ARIMA model can be extended adding additional (exogenous) inputs I, such a model is called ARIMAX, the same the SARIMA model becomes a SARIMAX model. Thus, the equation 5.6 becomes:

$$\Phi_P(B^s)\phi(B)\nabla_s^D \nabla^d x_t = \Gamma I_t + \alpha + \Theta_Q(B^s)\theta(B)e_t \tag{5.9}$$

ARIMA (and SARIMA) modelling is used by many as a sophisticated benchmark for evaluating alternative proposals. Various implementations of ARIMA/ARIMAX models for STLF has been described in literature, see Cho et al. [43] and Fan and McDonald [72], and the review by Hagan and Behr [96].

5.2 SHORT-TERM LOAD FORECASTING

Short-term load forecasting is the forecasting of energy demand usually from one hour to one week. For this kind of problem various factors should be considered, such as weather data or, more in general, all the factors influencing the load/consumption pattern. This means that for an accurate load forecasting exogenous variables may be considered and they differ according to customer type: residential, commercial, and industrial.

Load Forecasting should be considered a critical task for the management, scheduling and dispatching in power systems, and it concerns with the prediction of energy demand in different time spans. Especially for the future energy networks (smart grids), to achieve a greater control and flexibility than in actual electric grids, a reliable forecasting of load demand could help to avoid dispatch problems given by unexpected loads, and give vital information to make decisions on power generation and purchase, especially with energy markets becoming more and more competitive. Furthermore, accurate prediction would have a significant impact on operation management, e.g. preventing overloading and allowing an efficient energy storage. In fact, in an environment where the fluctuations of energy market may strongly influence the energy consumption (e.g. time-based pricing), forecasting the demand, using all the information provided by metering and sensing technologies, is vital in order to have an effective management of peaks (Demand Response).

The ability of predict future behaviours and energy demand is part of the intelligence required by future distribution networks, where information technology will be strongly applied (see EU ADDRESS project [1]). An intensive use of Distributed Generation raise new challenges, such the need of a 'distributed intelligence' in order to deal with data originated in diverse places and performing effective choices in a dynamic environment. An example of this new scenario is presented in Vale et al. [219] where various optimization heuristics are applied to economic dispatch problem in smart grids. Similar considerations can be made whether a smaller scale is considered: predict future situations may be critical for Energy Management Systems (EMS) (if we consider generation or transmission systems) or Building Management Systems (BMS) for single buildings.

Many excellent results in real applications with neural networks to STLF have been presented, the work of [132] is a good example. To make easier the comparison between traditional forecasting methods and less traditional ones (e.g. neural networks), ASHRAE has organized two interesting benchmarks in 1993 and 1996 [183, 139, 95].

A review of the application of neural networks to STLF can be found in Hippert et al. [103] and Metaxiotis et al. [167]. In this thesis a survey about the application of various computational intelligence techniques to building energy forecasting can be found in Section 2.3.

5.3 Application to Real Data

In the following sections we present an application of linear and non-linear models to the problem of short-term load forecasting using real data from an office building.

Load data usually exhibits seasonality, sometimes showing more than one periodicity: in fact the load at a given moment may be dependent on the load on the previous hour but also on the previous day and so on.

A good forecasting has to be accurate and, very important, it must have a maximum error as low as possible. In fact, the effectiveness of an energy management system (EMS) may be strongly affected by the error peaks and a predictor with a low variance might be preferred to a predictor with a better average error but with higher error peaks. On Smart Grids, underestimating the energy demand may have a negative impact on the Demand Response and it makes the control of overload conditions harder. On the other hand, an overestimation may creates an unexpected surplus of production. In both cases, higher is the estimation error and higher are the managing costs involved, e.g. an energy district could be forced to buy energy from the grid at higher costs than it would have in case of a better prediction.

5.3.1 Real Data

The following data, used in this work, has been collected in ENEA Casaccia Research center near Rome, in Italy:

1. Electricity hourly load data: measured in one of the building of the center, named C59, taking into account lighting, air conditioning/heating and appliances.

2. Weather data: ambient temperature and solar irradiance measured with a sensor inside the center.

3. Occupancy data: number of people inside the C59 building has been estimated using the data provided by the badge readers at the main entrance of the center.

All the data has been collected for an entire year but for this work we decided to focus on a period of 13 weeks, starting from 1/9/2009 to 31/11/2009, for a total of 2184 hourly samples[1]. In the period considered the load has an

[1]Datasets are available at `http://dia.uniroma3.it/~defelice/download.html`

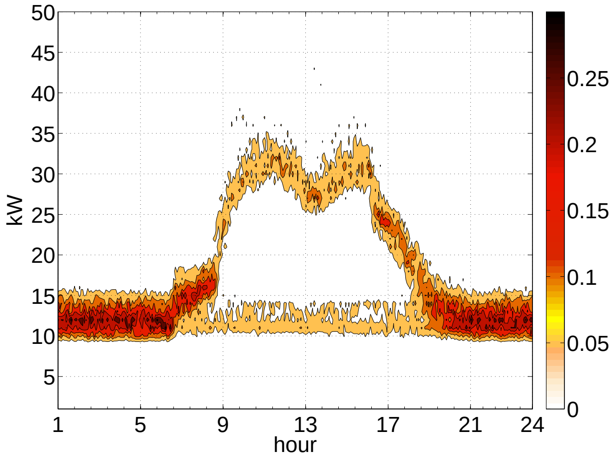


Figure 5.2: Data distribution during the day, the color displays the frequency of the load at a specific hour of the day

average value of 17.43 kW, a minimum of 8.58 kW and a maximum of 54.9 kW.

As expected, load data presents a clear periodicity which is summarized by the autocorrelation plot in Figure 5.3, where a weekly seasonality is stronger than the daily one.

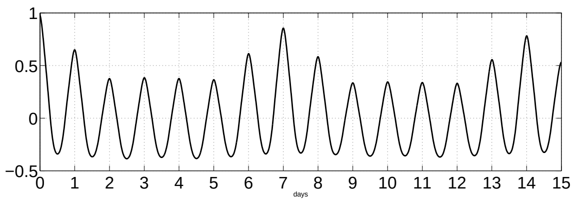


Figure 5.3: Autocorrelation Function (ACF) of the signal limited to 15 days

Number of people inside the C59 building during the working time normally varies from 45 to 70. Relation between occupancy and energy demand is shown in Figure 5.4.

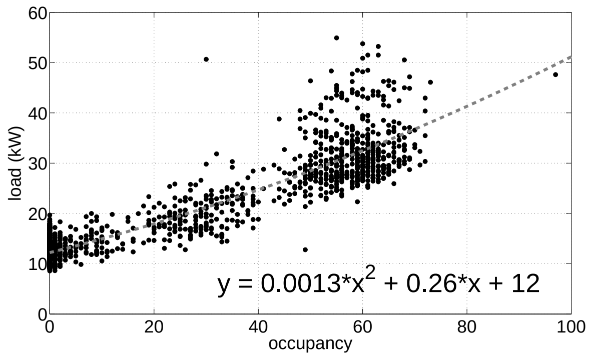


Figure 5.4: Relation between building occupancy and energy load. The grey line is the interpolation provided by a quadratic curve with the equation shown.

Energy demand is commonly influenced by weather, due to the effects of HVAC systems and lighting. In the C59 building the heating system is independent from the external temperature and no evident correlation (like for the occupancy) may be observed.

5.4 Forecasting Models

In this section we provide a brief description of the models involved in this work.

5.4.1 Naive model

In order to perform a meaningful comparison for the forecasting, a naive model should be introduced in order to quantify the improvement given by more intelligent and complex forecasting techniques. For seasonal data a naive model might be defined as:

$$x_t = x_{t-S} \tag{5.10}$$

with S the appropriate seasonality period. This model gives a prediction at time t presenting the value observed exactly a period of S steps before. For this work, after the considerations of the previous section, we put the value of $S = 168$ which corresponds to a week given that the data considered is hourly data.

5.4.2 Box-Jenkins models

Given the seasonality of the used data (as described in section 5.3.1) the seasonal variant of the ARIMA models, called SARIMA, has been chosen (see Section 5.1.3).

In this paper the selection process for the values of p, d, q, P, D and Q has been performed with the method proposed by Hyndman and Khandakar [111], implemented with R software [191], which explores the model space selecting the best model via AIC (Akaike Information Criterion) measure.

The model obtained and used in this work has the following orders: $p = 2, d = 1, q = 1, P = 1, D = 1, Q = 1$ (see Eq. 5.6).

5.4.3 Neural Networks

After a set of preliminary tests where we tested several inputs lags combinations, we selected the lags L corresponding to the previous 24 hours plus one-week and two-weeks lags, thus having $L = [1, 2, \dots, 24, 168, 336]$. The resulting NN model is the following:

$$x_t = f(x_{t-1}, \dots, x_{t-24}, x_{t-168}, x_{t-332}) \tag{5.11}$$

As in Eq. 5.2 the model with external data is obtained adding to the previous the information with the same lags, $I_t, I_{t-1}, \dots, I_{t-24}, I_{t-168}, I_{t-332}$.

In our work we selected two typologies of feed-forward neural networks: multi-layer perceptrons (MLPs) and radial-basis function networks (RBFs).

In the MLP design phase, after a set of preliminary tests, we have chosen the value of 64 hidden neurons[2]. The activation functions has been set to an hyperbolic tangent for the hidden layer and linear transfer function for the output layer. Finally, the chosen training algorithm is the Levenberg-Marquardt back-propagation with the initial value of the damping parameter set to 0.1. Training algorithm stop criteria has been set to the reaching of 1000 epochs.

For the RBFs we proceeded in the same way for the selection of hidden neurons as for MLPs, opting for networks with 128 Gaussian basis functions into the hidden layer. The training part consists of a random selection of input samples as initial coordinates of the hidden functions and then a scaled-conjugate gradient (SCG) algorithm for the optimization of weights and function coordinates.

[2]this value is near to the one suggested by the rule of thumb which states that $\mathrm{n.neurons} = \frac{\mathrm{n.inputs}}{2} + \sqrt{\mathrm{dataset\ size}}$

5.5 Neural Networks Ensembles

In this work two kinds of neural networks ensembles are examined: a simple averaging one and one based on the negative correlation among errors.

The first one is the simplest way to combine M neural networks, an arithmetic mean of their outputs (y_i) performed as:

$$y_{ens}(x_k) = \frac{1}{M}\sum_{i=1}^{M} y_i(x_k) \tag{5.12}$$

In this method all the NNs are trained separately and then, for each input x_k, the ensemble output is computed with an average of all the outputs of the NNs within the ensemble. The computation of this ensemble is performed on a set of 100 MLP NNs trained as described in Section 5.4.3

The second ensembling method considered in this paper is called Regularized Negative Correlation Learning (RNCL) and it has been proposed by Chen and Yao in [42]. This method improves the Negative Correlation Learning (NCL) adding a regularization term with the objective of reduce the overfitting problem. Regularization helps the network to avoid overfitting, i.e. improving generalization, penalizing large weights which may lead to rough outputs. In this work RBF Neural Networks are used for the RNCL ensemble, preferred to MLP networks for computational reasons.

5.6 STLF Univariate Approach

In this section we apply the selected models, built using only a part of the available data, on the prediction of hourly load on the remaining part of the dataset not used for the calibration/training.

The building hourly energy consumption dataset has been divided in two parts (see Figure 5.5): the first consists of 12 weeks (2016 hourly samples) and has been used for the models calibration (training) and the last one, consisting of a single week (168 samples), is the testing/validation part used for the evaluation of the models performances.

The calibration/training of the models has been performed on the first part and then 1-24 hours ahead prediction is carried out on the remaining part of the dataset (see Figure 5.6), obtaining at the end of the test phase 145 prediction windows (the number of intervals of 24 hours within the testing part).

The testing part has been subdivided in two equal parts (`T1` and `T2`) in order to provide more detailed information on the behaviour of the forecasting

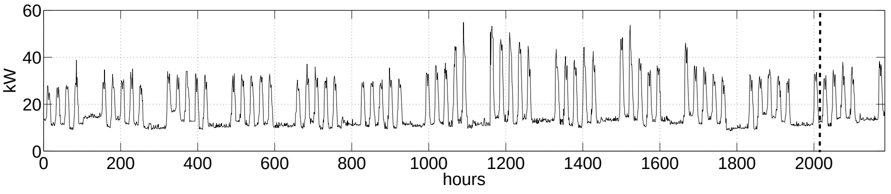


Figure 5.5: Dataset used for the comparison, the dashed line separates training from the testing part.

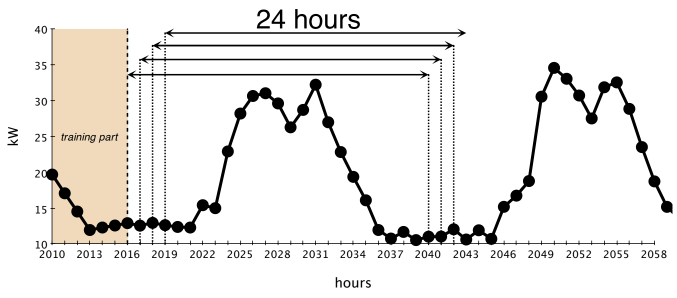


Figure 5.6: During the testing phase a 24-hours prediction is performed on the testing part of the dataset

models.

We considered a forecasting method where the predicted value at time t is used as input for the successive prediction at time $t+1$ and this for time $t+2$ and so on, this is a common approach for the Box-Jenkins models and thus we used the same for NNs (see Section 5.1.1).

Two performance criteria such as MSE and MAE were used to compare the selected models, their formula is given below:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{5.13}$$

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{5.14}$$

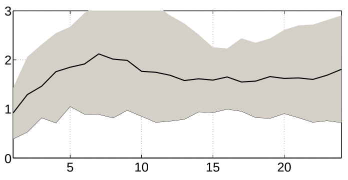
In addition, the maximum absolute error is considered given its critical importance for the STLF problem as discussed in Section 5.3.

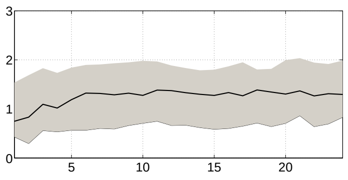
In Table 5.1 the performances of the selected models are shown, both on the training and the testing parts of the dataset.

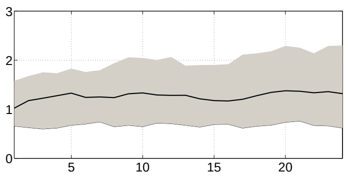
Table 5.1: Models performances: univariate approach (all the values are rounded to two decimals). In brackets the standard deviation where needed and in **bold** the best model error for the testing part.

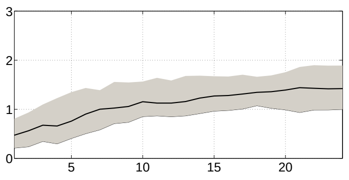
	Training		Testing T1			Testing T2		
Model	MAE	MSE	MAE	MSE	Max	MAE	MSE	Max
Naive	2.45	14.97	2.11	7.61	7.35	2.28	6.4	6.36
SARIMA	1.14	4.34	1.89	5.52	**7.15**	1.24	2.17	4.79
MLP best training	0.67	1.41	2.06	9.1	23.95	2.22	13.55	24.11
Average MLP	1.03 (0.38)	3.65 (2.19)	2.34 (0.79)	10.9 (17.88)	12.53 (9.22)	2.49 (1.47)	21.67 (59.29)	15.69 (15.2)
RBF best training	1.17	3.36	1.72	4.36	8.15	1.21	**2.14**	**4.51**
ANN RBF	3.11	29.51	2.96	19.44	12.49	3.20	19.91	15.39
RNCL	1.51	5.98	1.47	3.34	7.28	**1.07**	2.82	8.53

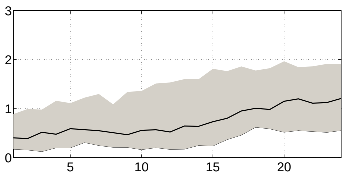
As expected, all the proposed models give better results than the naive model. MLP neural networks show a high variance in their performances and in order to obtain a more significant error measure, we can use the median error, which is less sensitive to outliers, instead of the mean error: the average MAE becomes 2.13 and 2.21, respectively for T1 and T2 and the MSE becomes 7.27 and 8.25. It's worth noting that the lowest testing MAE for the MLPs is 1.36 and 0.91, respectively for T1 and T2, and the lowest testing MSE is 2.99 and 1.78, values very near to the errors obtained by the MLP ensemble. For the RNCL ensemble the situation is slightly different: the best testing MAE is 1.58/1.61 and the MSE is 3.88/4.22, values definitely higher than the ones made by the ensemble. We can observe how RBF networks achieve an higher


(a) SARIMA T1


(b) MLP Ensembling T1


(c) RNCL T1


(d) SARIMA T2


(e) MLP Ensembling T2

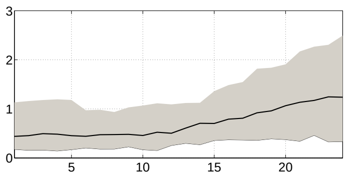
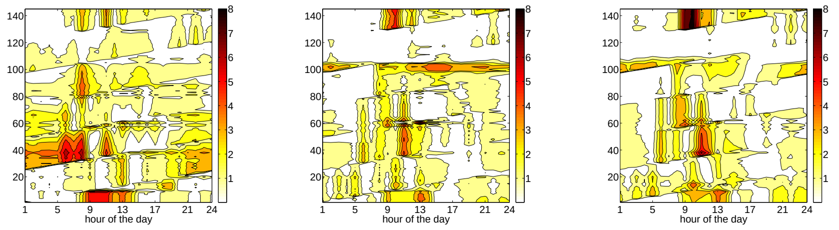

(f) RNCL T2

Figure 5.7: Univariate approach: 24-hours ahead forecasting absolute errors on both T1 and T2. In light grey the area between the 1st and the 3rd quartiles.

average error and standard deviation than MLP networks and this difference is reflected by the performance of their respective ensembles. The SARIMA model, despite its average results are higher than ANN ensembles, achieves a low maximum absolute error in T1 and near to the best one in testing T2.

As is evident in Figure 5.7, the numbers shown in the Table 5.1 are not able to capture all the differences between behaviours of the various models. In this figure the errors for each look-ahead and, more important, the variance of such errors are shown. In general, we can observe a degradation of performance with higher look-ahead horizons, and the testing part T1 seems to be more challenging than the other part, which lead to lower absolute errors because a weekend is involved (see Figure 5.5).



(a) SARIMA (b) MLP Ensembling (c) RNCL

Figure 5.8: Absolute errors (in kW) made during testing parts T1 and T2 for the univariate approach models. On the Y axis there are the various forecasting windows and on the X axis the hour of the day of each of the 24 prediction errors. Note that the color scale is not the same in each plot.

Figure 5.8 gives an overview of the forecasting errors during the testing weeks (considering both T1 and T2). Obviously there are some part of the testing data where all the models show low accuracy in their prediction, and as expected higher errors are present during the start of the working time (around 8 AM) and during lunchtime (12-13).

5.7 STLF with External Data

In this section we investigate how the forecast of the selected models is affected by the introduction of external data, e.g. building occupancy. As discussed in section 5.3.1, for the particular problem we are coping with, the introduction of weather data might not improve the forecasting and so they have not been used.

In this section, three kind of external data has been considered:

1. Information about the hour of the day
2. A 1/0 flag whether the sample has been collected in a workday or not
3. Building occupancy

The calculation of the first two kind of external data is trivial but for the building occupancy we should need to predict the value for the same interval of the load forecast. In this work we make the assumption that we will be able to know the exact value of such data in the future, otherwise we would have to forecast that value and then using the predicted value to perform the electric load forecast.

Experimental settings are the same used in the previous section, we put for sake of comparison again the results of the naive model (see Section 5.4.1).

Table 5.2: Models performances: approach with occupancy, hour of the day and workday flag (all the error values are rounded to two decimals). In brackets the standard deviation where needed and in **bold** the best model error for the testing part.

	Training		Testing T1			Testing T2		
Model	MAE	MSE	MAE	MSE	Max	MAE	MSE	Max
Naive (no ext. data)	2.45	14.97	2.11	7.61	7.35	2.28	6.4	6.36
SARIMA	1.13	4.31	1.91	5.61	8.00	1.20	2.07	5.18
MLP best training	0.36	0.70	3.51	20.28	18.00	2.20	11.83	24.53
Average MLP	1.20 (0.31)	3.25 (1.52)	2.46 (0.83)	12.13 (16.8)	13.84 (16.62)	2.34 (1.00)	11.61 (10.61)	13.00 (6.01)
MLP Ensemble	0.74	1.47	1.42	3.30	7.98	**0.75**	**1.27**	4.79
ANN RBF best training	1.06	2.48	1.36	3.03	6.43	0.88	1.61	7.05
Average ANN RBF	1.65	7.71	1.97	7.99	8.22	1.77	8.98	10.74
RNCL	1.15	3.35	**1.33**	**2.71**	**5.37**	0.92	1.62	**4.52**

According to Table 5.2 all the models, except SARIMA, improves their performance in at least one of the two testing parts. The testing part T2 exhibits the most evident error reduction due to the introduction of external data, MLP ensemble reduced its MAE from 1.09 to 0.75 and it has almost halved the MSE (from 2.4 to 1.27) and similar results have been obtained for RNCL. Differently from the results without external data, MLP ensemble exhibits a lower testing error than the best testing one among the MLP networks. In fact the best testing network shows a MAE of 1.37/0.98 and a MSE of 3.25/2.48. It's worth noting that the maximum error has been reduced respect to the errors shown in Table 5.1 for all the models, apart from SARIMA, in both the testing sets. RBF networks present a marked improvement, halving the MSE in both testing parts and clearly reducing both MAE and maximum errors.

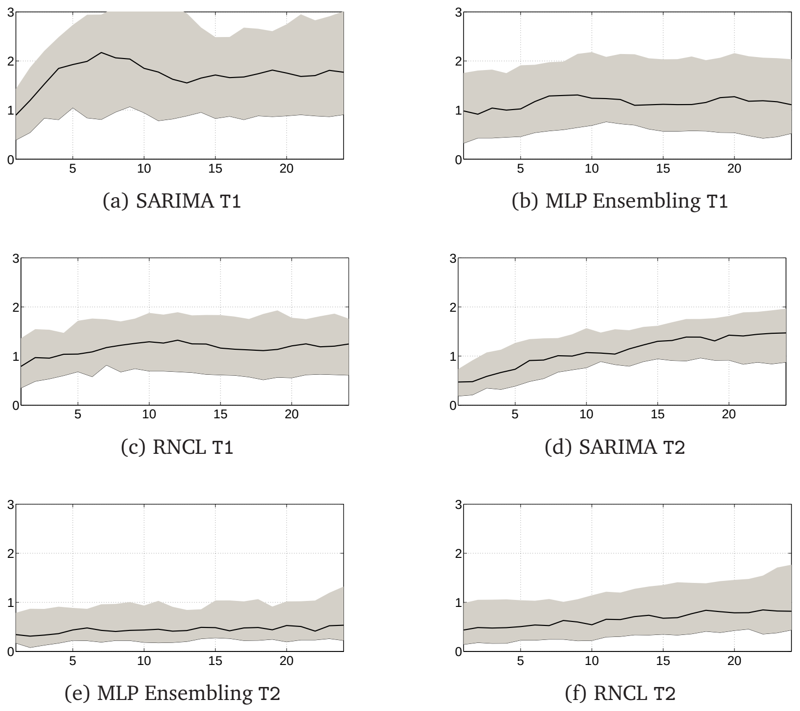
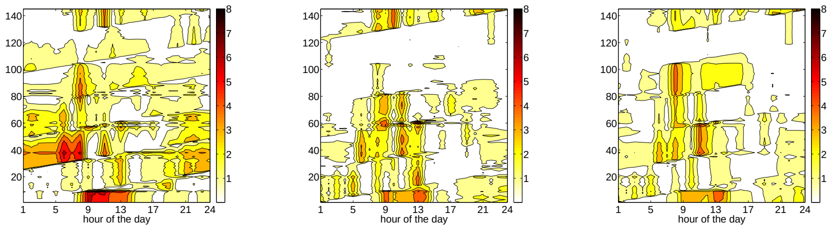

(a) SARIMA T1 (b) MLP Ensembling T1 (c) RNCL T1 (d) SARIMA T2 (e) MLP Ensembling T2 (f) RNCL T2

Figure 5.9: Forecasting with external data: 24-hours ahead forecasting absolute errors on both T1 and T2. In light grey the area between the 1st and the 3rd quartiles.

The average absolute error for the 24-hours prediction is shown in Figure 5.9 and with respect to Figure 5.7 the introduction of external data seems to have reduced the variance of the errors (the grey area) for the ANN-based models. Additional information is provided in Figure 5.10, where, after a comparison with Figure 5.8, we can see in which part of the dataset additional data has reduced the error.



(a) SARIMA (b) MLP Ensembling (c) RNCL

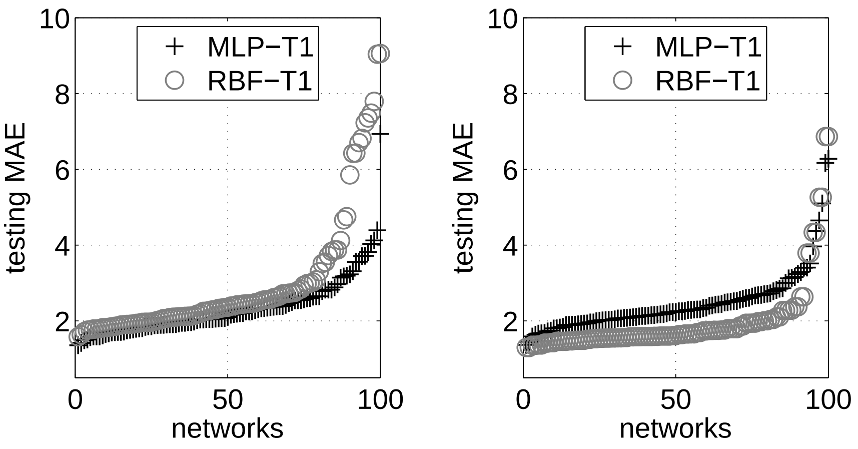
Figure 5.10: Absolute errors (in kW) made during testing parts T1 and T2 models with external data. On the Y axis there are the various forecasting windows and on the X axis the hour of the day of each of the 24 prediction errors. Note that the color scale is not the same in each plot.

5.8 Discussion

In Figure 5.11 testing absolute errors are shown, arranged in ascending order, for all the MLP and RBF networks on testing set T1 (we omitted T2 for sake of clearness). It's evident how neural network ensembles exhibits an error lower or at least equal than the best network, both for MLP and RBFs. This means that ensembling allows, thank to the exploitation of all the information 'contained' within the trained networks, to achieve an effective forecasting overcoming the drawbacks of neural networks: overfitting and the high variability of the performance of common training algorithms due to the their tendency to get stuck in local minima.

Another interesting observation, comparing Figures 5.11b and 5.11a, is that after the introduction of external data RBF networks trained with RNCL gives a more effective forecast than MLP networks, as it is shown also in Tables 5.1 and 5.2. More in general, the information provided by external data seems to help the non-linear models to improve their modelling, in Figure 5.12b and 5.12c is evident how the new information provided helped the forecast, e.g. the peak present at about time 100 in both the ANN-based models disappeared after the introduction of new data. Differently, the SARIMA model, which is linear, didn't show an improvement with the use of external data and indeed the difference of performance between it and the ANN-based models drastically increased in the second part of the experimentations.

Compared to a naive model both a time-series SARIMA model and neu-



(a) Without external data

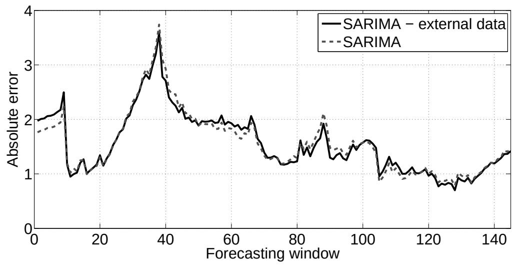
(b) With external data

Figure 5.11: Testing MAE for MLP and RBF networks

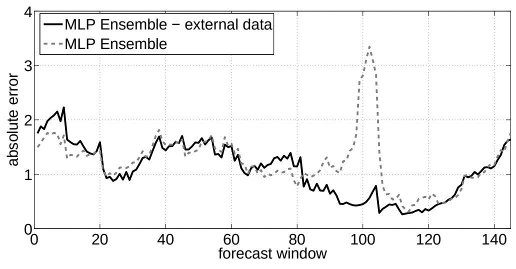
ral networks exhibit better performances and the latter have shown a drastic improvement when additional data is included in input.

The results underlines how creating an ensemble of neural networks consents to overcome a critical problem of such methods, the high variance of the performances, which may be a big limit for their applicability to engineering fields where the reliability (low variance of results) is commonly preferred to the overall average accuracy. Our results indicate that even a simple ensembling method as the arithmetic mean of the outputs allows to improve appreciably the error of the single neural networks.

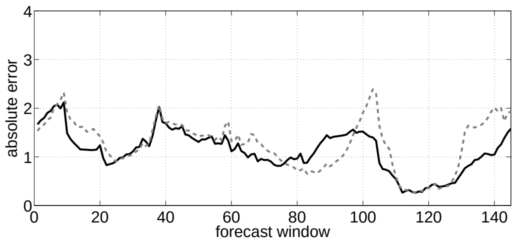
RNCL ensemble, and more in general RBF networks, shows the interesting ability of achieving a marked error reduction after the introduction of external data, a phenomenon that deserves further examination in future.



(a) SARIMA



(b) MLP Ensembling



(c) RNCL

Figure 5.12: Average absolute error for each 24-hours window: comparison between the SARIMA, MLP ensemble and RNCL models with and without the use of external data

CHAPTER 6

Conclusions and Future Work

This thesis work tries to summarize all the work done with my research group at ENEA during my Ph.D. on the challenging field of optimization and modellization of a variety of energy systems. The majority of the projects involved complex and noisy systems with a little knowledge about their dynamics, for this reason the choice of black-box methods was natural in many cases.

The following list is a brief summary of the novel contributions from this thesis:

- A review of the applications of computational intelligence in building energy systems is presented (Chapter 2): this is the first survey of this kind on this topic (previous work were focused on a smaller area of techniques)
- A technique to improve the effectiveness of neural networks training for modelling purposes has been presented in Chapter 3 and its application has been investigated on a particular modelling problem
- A fitness approximation technique has been applied in Chapter 4 in order to achieve in a reasonable amount of time the optimisation of the start-up phase of a combined-cycle heat power plant, in the same chapter a multi-objective evolutionary computational approach has been studied and experimented.
- Neural networks ensembles has been applied for the first time on a short-term load forecasting problem (Chapter 5) and their performances have been evaluated and investigated on real data.

All the Computational Intelligence (CI) paradigms have gained interests in the last decades for several reasons, one of them is surely their novelty and so the number of issue still open, but another important reason for their success is the possibility to apply them easily to any problem where data is available. In the case of optimization problems there is just another condition: the existence of a performance (fitness) function that allows to decide which solution is the best among a set of two or more.

The objective of Chapter 2 is to show the various application of computational intelligence to the emerging field of building energy systems optimization, control and forecasting. Although the survey examines scientific works ranging from 1995 to 2010, about a third of the considered works is concentrated in the last two years, giving an idea of the trend about the application of CI in this field. Another interesting point is the fact that among all the CI methodologies, neural networks are probably the most used, indeed in this survey almost half of the work reviewed uses them for modelling or prediction purposes. In fact, while in optimization problems the utilization of evolutionary computation is often an alternative to traditional methods (which in some cases shows similar performances), for modelling and forecasting purposes neural networks shows an incomparable easiness of application even in applications where traditional methods are inapplicable.

All the conclusions we can draw from the survey are in some way underlined in the following chapters, each focused on a specific application.

In Chapter 3, neural networks are used for modelling Italian ambient temperature for the purpose of giving a precise estimation to use inside a software simulator. An hybrid approach between a genetic and the back-propagation algorithm tries to overcome classical limitations of neural networks training, obtaining a better accuracy than common training methods. The interesting part lies not into the resulting errors, but in the simplicity with the two methods were combined to achieve a new algorithm. Moreover, the work presented in Chapter 3 underlines how it was possible to improve a well-known thermal energy systems simulator (TRNSYS) achieving a better accuracy in estimating thermal loads and thus heating costs.

Totally different is the application described in Chapter 4, which is about the optimization of a large combined cycle power plant. In that case both a single- and a multi-objective optimization algorithms are used, both based on Evolutionary Computation methods. The main issue was the difficulty to decide a single performance function, in fact a problem of this complexity has several factors to take into consideration, normally in contrast one with each other, e.g. the energy output with the pollutant emissions. Two strategies in fact were proposed: each of them with its advantages and drawbacks. It's obvious that the single-objective approach led to a simpler optimization framework than multi-objective approach, but in the first case a fuzzy logic-based representation of the knowledge of operators, gained through experience, has been used as fitness function, adding in that way a further 'layer' of complexity to the whole framework. In the multi-objective case we obtain at the end of the optimization a set of solutions instead of a single optimal one, an interesting feature of this class of optimization algorithms, but this is also the reason because a comparison between single- and multi-objective approach is not straightforward.

However, both the evolutionary algorithm-based approaches for the start-up optimization revealed how it is simple to set up an optimization framework, which incorporates problem-specific knowledge and various constraints, using a software simulator as performance function. Although we weren't able to experiment our solutions on the real plant (for a lot of practical reasons), the application presented is an evident example of the advantages of black-box optimization with evolutionary algorithms.

In Chapter 5 the last real-world application of this thesis is presented. A common practical problem of load forecasting, which is gaining importance year after year, is tackled with neural networks ensembles. The comparison with standard benchmarks highlights some interesting features of neural network ensembles: the easiness of exploitation of additional information and the low error variance (and so the reliability). The forecasting methodology proposed in this chapter and applied on a single building may be extended on several buildings with a larger number of additional inputs with a little effort, due to neural network modelling structure. The use of ensemble, which represent the novelty of this specific application, allowed to reduce the error variance implicit by the classical training algorithms for neural networks, making this approach an interesting candidate for a realistic industrial application.

6.1 Discussion and Future Work

All the works presented in this thesis have shown some interesting (and working) application of computational intelligence to energy systems. As all the good research works, they have raised as many questions as the answers provided.

The first consideration is about the existing diverse computational intelligence techniques. Month after month, a large amount of new articles about these methods are published on scientific journal and presented on international conferences, many of them presenting new algorithms and variants of existing algorithms. When approaching a real-world problem the first question is the most critical: “what would be the best approach for this application?”. Many researchers in the last years are making order on the huge amount of algorithms and variants (and acronyms), trying to give best practises and suggestions, supported by theoretical studies and experimentations on benchmarks. For example, many researchers are still using Genetic Algorithms because of their presence on the most popular software packages (e.g. MATLAB) without considering other, sometimes simpler, meta-heuristics. It’s important to start with the simplest effective algorithm (e.g. Evolution Strategies for evolutionary computation optimization or back-propagation for neural networks) remembering the principle of Ockham’s Razor.

The use of valid benchmarks is a critical point for engineering works (like the ones presented in this thesis). In fact it’s very important to evaluate the performance of a technique comparing it with standard methodologies, in order to understand the effectiveness of the proposed method and, more important, to investigate the reason of its success (or failure). Moreover, if the first goal of an engineering research work is to propose a solution applicable in real-world problems, it’s fundamental to compare innovative and experimental solutions with the state-of-the-art, in other words comparing it with the first method an engineering would use to cope with that specific problem. For this reason, in the short-term load forecasting work at Chapter 5 we compared the proposed neural network ensemble-based forecasting methodology with both a traditional time series and a naive method and not only with another neural network-based method. For this reason, in the future, both the works presented in Chapter 3 and 4 will be extended with a more rigorous comparison.

The work on ambient temperature modelling is based on a hybrid approach with back-propagation networks and genetic algorithms, the comparison may be extended with traditional interpolation methods (e.g kriging) and other combinations for the hybrid approach (using Evolution Strategies instead of a GA or support vector machines instead of neural networks). Similarly, the application of evolutionary algorithm on the plant start-up optimization problem may be extended with other black-box optimization approaches like quasi-Newton approximation methods or other evolutionary algorithms, both single- and multi-objective.

Finally, the survey presented at Chapter 2 might be extended to larger energy systems, like power systems.

Appendix A

Computational Intelligence in Software Packages

This appendix gives a list of some of the most common software implementations of neural networks and computational intelligence algorithms.

Neural Networks

- MATLAB [http://www.mathworks.com/]: there are implementation of various typologies of neural networks (MLP, RBF, GRNN etc) with the Neural Network toolbox.

- R [http://www.r-project.org/]: the package nnet introduces feed-forward neural networks with a single hidden layer.

- WEKA [http://www.cs.waikato.ac.nz/ml/weka/]: this open-source project is a collection of data mining and classification algorithms, among them neural networks.

- SAS Enterprise Miner [www.sas.com/technologies/analytics/datamining/]: SAS software includes neural networks for the prediction and modelling part.

Evolutionary Computation

- MATLAB [http://www.mathworks.com/]: within the Optimization Toolbox there are a Genetic Algorithm and an implementation of NSGA-II. More evolutionary algorithms are present in GEATbx [`http://www.geatbx.com/`] toolbox. An implementation of GP for MATLAB is provided by GPLAB [`http://gplab.sourceforge.net/`].

- NEO Software [`neo.lcc.uma.es/software/index/`]: NEO (Networking and Emerging Optimization) group of Universidad De Malaga offers several types of Evolutionary Algorithms and metaheuristics.
- Optimization Algorithm Toolkit (OAT) [`optalgtoolkit.sourceforge.net/`]: this java-based open-source project presents various optimization algorithms and common benchmark problems with a nice graphic interface and some statistical tools.
- CMA-ES Source Code [`www.lri.fr/~hansen/cmaes_inmatlab.html`]: various implementations in several programming languages of CMA-ES algorithm are provided on Nikolaus Hansen webpage of Machine Learning and Optimization group (TAO) at INRIA.
- ParadisEO [`http://paradiseo.gforge.inria.fr/`]: this C++ framework implements various metaheuristics: among them evolutionary computation for single- and multi-objective problems.

Bibliography

[1] http://www.addressfp7.org/.

[2] www.neural-forecasting-competition.com.

[3] Visual doe 4.0 software. architectural energy cooperation, san francisco, usa, 2005.

[4] Paulo J.L. Adeodato, Adrian L. Arnaud, Germano C. Vasconcelos, Rodrigo C.L.V. Cunha, and Domingos S.M.P. Monteiro. Mlp ensembles improve long term prediction accuracy over single networks. *International Journal of Forecasting*, In Press, Corrected Proof:–, 2009.

[5] Turang Ahadi-Oskui, Stefan Vigerske, Ivo Nowak, and George Tsatsaronis. Optimizing the design of complex energy conversion systems by branch and cut. *Computers & Chemical Engineering*, 34(8):1226 – 1236, 2010.

[6] Mitra A.K. and Sankar N. Forecasting maximum temperatures via fuzzy nearest neighbour model over delhi. *Applied and Computational Mathematics*, 6(2):288–294, 2007.

[7] F. Alobaid, R. Postler, J. Ströhle, B. Epple, and H-G. Kim. Modeling and investigation start-up procedures of a combined cycle power plant. *Applied Energy*, 85(12):1173–1189, 2008.

[8] O. Altandombayci and M. Golcu. Daily means ambient temperature prediction using artificial neural network method: A case study of turkey. *Renewable Energy*, 34(4):1158–1161, april 2009.

[9] K. Andersen, G.E. Cook, K. Ramaswamy, and G. Karsai. Artificial neural networks applied to arc welding process modeling and control. In *Industry Applications Society Annual Meeting, 1989., Conference Record of the 1989 IEEE*, pages 2327 –2331 vol.2, October 1989.

[10] A.A. Argiriou, I. Bellas-Velidis, and C.A. Balaras. Development of a neural network heating controller for solar buildings. *Neural Networks*, (13):811–820, 2000.

[11] A.A. Argiriou, I. Bellas-Velidis, M. Kummert, and P. André. A neural network controller for hydronic heating systems of solar buildings. *Neural Networks*, (17):427–440, 2004.

[12] S. Atthajariyakul and T. Leephakpreeda. Neural computing thermal comfort index for hvac systems. *Energy Conversion and Management*, 46(15-16):2553 – 2565, 2005.

[13] T. Ayata, E. Arcaklıouglŭ, and Yıldız. Application of ann to explore the potential use of natural ventilation in buildings in turkey. *Applied Thermal Engineering*, (27), 2007.

[14] Merih Aydinalp, V. Ismet Ugursal, and Alan S. Fung. Modeling of the appliance, lighting, and space-cooling energy consumptions in the residential sector using neural networks. *Applied Energy*, 71(2):87 – 110, 2002.

[15] M. Aydinalp-Koksal and V. Ismet Ugursal. Comparison of neural network, conditional demand analysis, and engineering approaches for modeling end-use energy consumption in the residential sector. *Applied Energy*, (85):271–296, 2008.

[16] M. Aydinalp-Koksal, V. Ismet Ugursal, and A.S. Fung. Modeling of the space and domestic hot-water heating energy-consumption in the residential sector using neural networks. *Applied Energy*, (79):159–178, 2004.

[17] W. Azmy, N. El Gayar, A. Atiya, and H. El-Shishiny. Mlp, gaussian processes and negative correlation learning for time series prediction. In J. Benediktsson, J. Kittler, and F. Roli, editors, *Multiple Classifier Systems*, volume 5519 of *Lecture Notes in Computer Science*, pages 428–437. Springer Berlin / Heidelberg, 2009.

[18] Johannes Bader, Kalyanmoy Deb, and Eckart Zitzler. Faster hypervolume-based search using monte carlo sampling. In Matthias Ehrgott, Boris Naujoks, Theodor J. Stewart, and Jyrki Wallenius, editors, *Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems*, volume 634 of *Lecture Notes in Economics and Mathematical Systems*, pages 313–326. Springer Berlin Heidelberg, 2010.

[19] A. Balestrino, F. Bini Verona, and M. Santanche. Time series analysis by neural networks: Environmental temperature forecasting. *Automazione e Strumentazione*, 12(42):81–87, 1994.

[20] A.E. Ben-Nakhi and M.A. Mahmoud. Energy conservation in buildings through efficient a/c control using neural networks. *Applied Energy*, (73), 2002.

[21] A.E. Ben-Nakhi and M.A. Mahmoud. Cooling load prediction for buildings using general regression neural networks. *Energy Conversion & Management*, (45):2127–2141, 2004.

[22] I. Bertini, F. Ceravolo, M. Citterio, M. De Felice, B. Di Pietra, F. Margiotta, S. Pizzuti, and G. Puglisi. Ambient temperature modelling with soft computing techniques. *Solar Energy*, 84(7):1264 – 1272, 2010.

[23] I. Bertini, M. De Felice, F. Moretti, and S. Pizzuti. Start-up optimisation of a combined cycle power plant with multiobjective evolutionary algorithms. In *Applications of Evolutionary Computation*, volume 6025/2010 of *Lecture Notes in Computer Science*, pages 151–160. Springer Berlin / Heidelberg, 2010.

[24] N.V. Bhat, Jr. Minderman, P.A., T. McAvoy, and N.S. Wang. Modeling chemical process systems via neural computation. *Control Systems Magazine, IEEE*, 10(3):24 –30, April 1990.

[25] E. T. Bonataki and K.C. Giannakoglou. Preliminary design of optimal combined cycle power plants through evolutionary algorithms. In *Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems, EUROGEN 2005*, 2005.

[26] A.M. Boulos and K.J. Burnham. A fuzzy logic approach to accommodate thermal stress and improve the start-up phase in combined cycle

power plants. In *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, volume 216, pages 945–956, January 2002.

[27] George E. P. Box and Gwilym M. Jenkins. *Time series analysis; forecasting and control*. Holden-Day San Francisco, 1970.

[28] Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996. 10.1007/BF00058655.

[29] Peter J. Brockwell and Richard A. Davis. *Introduction to Time Series and Forecasting*. Springer, March 2002.

[30] F.M. Burnet. *The Clonal Selection Theory of Acquired Immunity*. Vanderbilt University Press, Nashville, T.N., 1959.

[31] B. Burton and R.G. Harley. Reducing the computational demands of continually online trained artificial neural networks for system identification and control of fast processes. In *Industry Applications Society Annual Meeting, 1994., Conference Record of the 1994 IEEE*, pages 1836 –1843 vol.3, October 1994.

[32] B. Burton, F. Kamran, R.G. Harley, T.G. Habetler, M.A. Brooke, and R. Poddar. Identification and control of induction motor stator currents using fast on-line random training of a neural network. *Industry Applications, IEEE Transactions on*, 33(3):697 –704, 1997.

[33] Y. Cai. Prediction of protein structural classes by support vector machines. *Computers & Chemistry*, 26(3):293–296, February 2002.

[34] L.G. Caldas and L.K. Norford. A design optimization tool based on a genetic algorithm. *Automation in Construction*, (11):173–184, 2002.

[35] L.G. Caldas and L.K. Norford. Genetic algorithms for optimization of building envelopes and the design and control of hvac systems. *Journal of Solar Energy Engineering*, 125:343, August 2003.

[36] F. Casella and F. Pretolani. Fast start-up of a combined-cycle power plant: a simulation study with modelica. In *Proceedings 5th International Modelica Conference*, pages 3–10, September 6-8 2006.

[37] C. Chang and C. Lin. Libsvm: a library for support vector machines.

[38] Y-C. Chang. Genetic algorithm based optimal chiller loading for energy conservation. *Applied Thermal Engineering*, (25):2800–2815, 2005.

[39] Y-C. Chang. Application of genetic algorithm to the optimal-chilled water supply temperature calculation of air-conditioning systems for saving energy. *International Journal of Energy Research*, (31):796–810, February 2007.

[40] Y-C. Chang. Optimal chiller loading by evolution strategy for saving energy. *Energy and Buildings*, (39):437–444, 2007.

[41] Y-C. Chang, J-K. Lin, and M-H. Chuang. Optimal chiller loading by genetic algorithm for reducing energy consumption. *Energy and Buildings*, (37):147–155, 2005.

[42] H. Chen and X. Yao. Regularized negative correlation learning for neural network ensembles. *IEEE Transactions on Neural Networks*, 20(12):1962–1979, 2009.

[43] M.Y. Cho, J.C. Hwang, and C.S. Chen. Customer short term load forecasting by using arima transfer function model. volume 1, pages 317–322, nov. 1995.

[44] M.H. Choueiki, C.A. Mount-Campbell, and S.C. Ahalt. Building a 'quasi optimal' neural network to solve the short-term load forecasting problem. *Power Systems, IEEE Transactions on*, 12(4):1432 –1439, nov 1997.

[45] T.T. Chow, G.Q. Zhang, Z. Lin, and C.L. Song. Global optimization of absorption chiller system by genetic algorithm and neural network. *Energy and Buildings*, (34):103–109, 2002.

[46] J.A. Clarke. *Energy simulation in building design*. Butterworth-Heinemann, 2001.

[47] Carlos A. Coello. An updated survey of ga-based multiobjective optimization techniques. *ACM Comput. Surv.*, 32:109–143, June 2000.

[48] D.A. Coley and J.A. Crabb. An artificial intelligence approach to the prediction of natural lighting levels. *Building and Environment*, 32(2):81–85, 1997.

[49] V. Congradac and F. Kulic. Hvac system optimization with co$_2$ concentration control using genetic algorithms. *Energy and Buildings*, (41):571–577, 2009.

[50] G.E. Cook, R.J. Barnett, K. Andersen, and A.M. Strauss. Weld modeling and control using artificial neural networks. *Industry Applications, IEEE Transactions on*, 31(6):1484 –1491, 1995.

[51] T.M. Cover. Geometrical and statistical properties of systems in linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, 14:326–334, 1965.

[52] Drury B. Crawley, Linda K. Lawrie, Frederick C. Winkelmann, W. F. Buhl, Y. Joe Huang, Curtis O. Pedersen, Richard K. Strand, Richard J. Liesen, Daniel E. Fisher, Michael J. Witte, and Jason Glazer. Energyplus: creating a new-generation building energy simulation program. *Energy and Buildings*, 33(4):319 – 331, 2001.

[53] D. Dasgupta and S. Forrest. Artificial immune systems in industrial applications. In *Proceedings of the IPMM '99*, page 257, 1999.

[54] M. De Felice and X. Yao. Neural networks ensembles for short-term load forecasting. In *IEEE Symposium Series in Computational Intelligence 2011 (SSCI 2011)*, 2011.

[55] Kenneth A. De Jong. *Evolutionary Computation: A Unified Approach*. The MIT Press, 1st edition, March 2002.

[56] K Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 2001.

[57] K. Deb, S. Agrawal, A. Patrap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In *Proceedings of the Sixth Parallel Problem Solving in Nature Conference*, pages 849–858, 2000.

[58] S.J. Dickinson and A. Bradshaw. Genetic algorithm optimisation and scheduling for building heating systems. In *Genetic Algorithms in Engineering Systems: Innovations and Applications, 1995. GALESIA. First International Conference on (Conf. Publ. No. 414)*, pages 106 –111, 12-14 1995.

[59] R. Dobrowolski, A. Witkowski, and R. Leithner. Simulation and optimization of power plant cycles. In *Proceedings of the 15th International Conference on Efficiency, Costs, Optimization, Simulation and Environmental Impact of Energy Systems*, pages 776–772, 2002.

[60] Robert H. Dodier and Gregor P. Henze. Statistical analysis of neural networks as applied to building energy prediction. *Journal of Solar Energy Engineering*, 126(1):592–600, 2004.

[61] Ö.A. Dombaycı. The prediction of heating energy consumption in a model house by using artificial neural networks in denizli–turkey. *Advances in Engineering Software*, (41):141–147, 2010.

[62] M. Dorigo. *Optimization, Learning and Natural Algorithms*. PhD thesis, Politecnico di Milano, Italy, 1992.

[63] H. Drucker, Chris, B. L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. In *Advances in Neural Information Processing Systems*, volume 9, pages 155–161. 1997.

[64] Russell C Eberhart, James Kennedy, and Yuhui Shi. *Swarm intelligence*. Morgan Kaufmann series in evolutionary computation. Elsevier, Burlington, MA, 2001.

[65] Agoston E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer-Verlag, 2003.

[66] B.B. Ekici and U.T. Aksoy. Prediction of building energy consumption by using artificial neural networks. *Advances in Engineering Software*, (40):352–362, 2009.

[67] H. K. Elminir, F. F. Areed, and T. S. Elsayed. Estimation of solar radiation components incident on helwan site using neural networks. *Solar Energy*, 79(3):270–279, September 2005.

[68] Andries P. Engelbrecht. *Computational Intelligence: An Introduction*. Wiley Publishing, 2007.

[69] D. G. Erbs, S. A. Klein, and W. A. Beckman. Estimation of degree-days and ambient temperature bin data from monthly-average temperatures. *ASHRAE journal*, pages 60–65, 1983.

[70] Scott E. Fahlman and Christian Lebiere. Advances in neural information processing systems 2. chapter The cascade-correlation learning architecture, pages 524–532. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[71] Ron Patton Faisel, Vasile Palade, Ron J. Patton, Joseba Quevedo, and S. Daley. Fault diagnosis of an industrial gas turbine using neuro-fuzzy methods. In *Preprints of the 15 th IFAC World Congress*, 2002.

[72] J.Y. Fan and J.D. McDonald. A real-time implementation of short-term load forecasting for distribution power systems. *IEEE Transactions on Power Systems*, 9(2):988–994, may 1994.

[73] P. O. Fanger. Thermal comfort. analysis and applications in environmental engineering,. 1970.

[74] Eugene A. Feinberg and Dora Genethliou. Load forecasting. In J.H. Chow, F.F. Wu, and J.J. Momoh, editors, *Applied Mathematics for Restructured Electric Power Systems: Optimization, Control and, Computational Intelligence*, pages 269–285. Springer, 2005.

[75] P.M. Ferreira, E.A. Faria, and A.E. Ruano. Neural network models in greenhouse air temperature prediction. *Neurocomputing*, (43):51–75, 2002.

[76] B.M. Flax. Intelligent buildings. *Communications Magazine, IEEE*, 29(4):24 –27, apr. 1991.

[77] L.J. Fogel. Autonomous automata. *Industrial Research*, (4):14–19, 1962.

[78] K. F. Fong, T. T. Chow, and V. I. Hanby. Development of optimal design of solar water heating system by using evolutionary algorithm. *Journal of Solar Energy Engineering*, 129(4):499–501, 2007.

[79] K.F. Fong, V.I. Hanby, and T.T. Chow. Hvac system optimization for energy management by evolutionary programming. *Energy and Buildings*, (38):220–231, 2006.

[80] K.F. Fong, V.I. Hanby, and T.T. Chow. System optimization for hvac energy management using the robust evolutionary algorithm. *Applied Thermal Engineering*, (29):2327–2334, 2009.

[81] C. M. Fonseca and P.J Fleming. Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, 1993.

[82] S. Forrest, A.S. Perelson, and R. Cherukuri. Self-nonself discrimination in a computer. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 202–212, 1994.

[83] Yoav Freund and Robert E. Schapire. Experiments with a New Boosting Algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.

[84] N. Ghrab-Morcos. Cheops: a simplified tool for thermal assessment of mediterranean buildings in both hot and cold seasons. *Energy and Buildings*, 37(6):651–662, 2005.

[85] Kai Goebel and Alice M. Agogino. Fuzzy sensor fusion for gas turbine power plants. volume 3719, pages 52–61. SPIE, 1999.

[86] P.A. Gonzalález Lanza and J.M. Zamarreño Cosme. A short-term temperature forecaster based on a state space neural network. 5(15):459–464, 2002.

[87] Pedro A. González and Jesús M. Zamarreño. Prediction of hourly energy consumption in buildings based on a feedback artificial neural network. *Energy and Buildings*, 37(6):595 – 601, 2005.

[88] P. Goovaerts. Geostatistical approaches for incorporating elevation into the spatial interpolation of rainfall. *Journal of Hydrology*, 228(1-2):113–129, February 2000.

[89] Antonio Augusto Gorni. The application of neural networks in the modeling of plate rolling processes. *Journal of Materials - Electronic J.O.M.-E*, 1997.

[90] L. Gosselin, M. Tye-Gingras, and F. Mathieu-Potvin. Review of utilization of genetic algorithms in heat transfer problems. *International Journal of Heat and Mass Transfer*, (52):2169–2188, 2009.

[91] M.M. Gouda, S. Danaher, and C.P. Underwood. Application of an artificial neural network for modelling the thermal dynamics of a building's

space and its heating system. *Mathematical and Computer Modelling of Dynamical Systems,* (8):333–344, 2002.

[92] Simulation Research Group. *DOE-2 Supplement — Version 2.1E*. Lawrence Berkeley National Laboratory,, lbl-34946 edition, 1993.

[93] A. Guillemin and N. Morel. An innovative lighting controller integrated in a self-adaptive building control system. *Energy and Buildings*, 33(5):477 – 487, 2001.

[94] S. R. Gunn. Support vector machines for classification and regression. Technical report, Faculty of Engineering, Science and Mathematics School of Electronics and Computer Science, May 1998.

[95] J.S. Haberl and S. Thamilseran. The great energy predictor shootout. ii : Measuring retrofit savings. *ASHRAE journal*, 40(1):49–56, 1998.

[96] Martin T. Hagan and Suzanne M. Behr. The time series approach to short term load forecasting. *IEEE Transactions on Power Systems*, 2(3):785–791, aug. 1987.

[97] Martin T. Hagan, Howard B. Demuth, and Mark Beale. *Neural network design*. PWS Publishing Co., Boston, MA, USA, 1996.

[98] M.T. Hagan and M.B. Menhaj. Training feedforward networks with the marquardt algorithm. *Neural Networks, IEEE Transactions on*, 5(6):989 –993, November 1994.

[99] L.K. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, oct. 1990.

[100] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[101] Babak Hassibi, David G. Stork, and Stork Crc. Ricoh. Com. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in Neural Information Processing Systems 5,* pages 164–171. Morgan Kaufmann, 1993.

[102] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 2nd edition, 1998.

[103] H.S. Hippert, C.E. Pedreira, and R.C. Souza. Neural networks for short-term load forecasting: Neural networks for short-term load forecasting: A review and evaluation. *IEEE Transaction on Power Systems*, 16(1):44, February 2001.

[104] Yu-Chi Ho. An explanation of ordinal optimization: soft computing for hard problems. *Inf. Sci.*, 113(3-4):169–192, 1999.

[105] J. Hofierka, J. Parajka, H. Mitasova, and L. Mitas. Multivariate interpolation of precipitation using regularized spline with tension. *Transactions in GIS*, 6(2):135–150, 2002.

[106] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, Michigan, 1975.

[107] Chich-Yi Huang, Tien-Chi Chen, and Ching-Lien Huang. Robust control of induction motor with a neural-network load torque estimator and a neural-network identification. *Industrial Electronics, IEEE Transactions on*, 46(5):990 –998, October 1999.

[108] W. Huang and H.N. Lam. Using genetic algorithms to optimize controller parameters for hvac systems. *Energy and Buildings*, (26):277–282, 1997.

[109] T. Huld, M. Šúri, E. Dunlop, and F. Micale. Estimating average daytime and daily temperature profiles within europe. *Environmental Modelling & Software*, 21(12):1650–1661, December 2006.

[110] A.J. Hunter. A climatic model for the prediction of percentile statistics for ambient temperature. *Monthly Weather Review*, 109(4):722–728, April 1981.

[111] Rob J. Hyndman and Yeasmin Khandakar. Automatic time series forecasting: the forecast package for r. Monash Econometrics and Business Statistics Working Papers 6/07, Monash University, Department of Econometrics and Business Statistics, June 2007.

[112] Rob J. Hyndman and Anne B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679 – 688, 2006.

[113] R.L. Iman, J.E. Campbell, and J.C. Helton. An approach to sensitivity analysis of computer models. i - introduction, input, variable selection and preliminary variable assessment. *Journal of Quality Technology*, 13:174–183, July 1981.

[114] Fluent Inc. *Fluent 6.3 users guide*, September 2006.

[115] N. Islam, C. Doscher, and T. Davies. Estimating missing daily maximum and minimum temperatures for mount cook, south island, new zealand, using a statistical model and 'ainet' neural network models. *Journal of Hydrology (New Zealand)*, 2(39):83–106, 2001.

[116] S. Jeffrey, J. Carter, K. Moodie, and A. Beswick. Using spatial interpolation to construct a comprehensive archive of australian climate data. *Environmental Modelling & Software*, 16(4):309–330, June 2001.

[117] Y. Jiang. Prediction of monthly mean daily diffuse solar radiation using artificial neural networks and comparison with other empirical models. *Energy Policy*, 36(10):3833–3837, October 2008.

[118] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9(1):3–12, January 2005.

[119] W. Jolly, J. Graham, A. Michaelis, R. Nemani, and S. Running. A flexible, integrated system for generating meteorological surfaces derived from point sources across multiple geographic scales. *Environmental Modelling & Software*, 20(7):873–882, July 2005.

[120] Iebeling Kaastra and Milton Boyd. Designing a neural network for forecasting financial and economic time series. *Neurocomputing*, 10(3):215 – 236, 1996. Financial Applications, Part II.

[121] S. Kajl, M-A. Roberge, L. Lamarche, and P. Malinowski. Evaluation of building energy consumption based on fuzzy logic and neural networks applications. 1997.

[122] S. A. Kalogirou. Long-term performance prediction of forced circulation solar domestic water heating systems using artificial neural networks. *Applied Energy*, (66):63–74, 2000.

[123] S. A. Kalogirou. Artificial neural networks and genetic algorithms in energy applications in buildings. *Advances in Building Energy Research*, 3:83–120, 2009.

[124] S. A. Kalogirou and M. Bojic. Artificial neural networks for the prediction of the energy consumption of a passive solar building. *Energy*, (25):479–491, 2000.

[125] S. A. Kalogirou and S. Panteliou. Thermosiphon solar domestic water heating systems: long-term performance prediction using artificial neural networks. *Solar Energy*, 69(2):163–174, 2000.

[126] J.H. Kämpf and D. Robinson. A hybrid cma-es and hde optimisation algorithm with application to solar energy potential. *Applied Soft Computing*, (9):738–745, 2009.

[127] J.H. Kämpf and D. Robinson. Optimisation of building form for solar energy utilisation using constrained evolutionary algorithms. *Energy and Buildings*, In Press, Corrected Proof, 2009.

[128] G. Kayo and R. Ooka. Building energy system optimizations with utilization of waste heat from cogenerations by means of genetic algorithm. *Energy and Buildings*, (In Press), 2010.

[129] Vojislav Kecman. *Learning and Soft Computing: Support Vector Machines, Neural Networks, and Fuzzy Logic Models*. MIT Press, Cambridge, MA, USA, 2001.

[130] J. Kennedy and R.C. Eberhart. Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks (ICNN '95)*, volume 4, pages 1992–1948, Perth, Western Australia, November-December 1995. IEEE Service Center.

[131] V. Khare, X. Yao, and K. Deb. Performance scaling of multi-objective evolutionary algorithms. In C. M. Fonseca, P.J Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO'03)*, volume 2632 of *Lecture Notes in Computer Science*, pages 376–390. Springer-Verlag, April 2003.

[132] A. Khotanzad, M. H. Davis, A. Abaye, and D. J. Maratukulam. An artificial neural network hourly temperature forecaster with applications in load forecasting. *IEEE Transaction on Power Systems*, 2(11):870–876, 1996.

[133] P.L.T. Kie and L.B. Theng. Intelligent control of heating, ventilating and air conditioning systems. In Köpppen et al., editor, *ICONIP 2008*, pages 927–934. Springer, 2009.

[134] S.A. Klein, W.A. Beckman, J.W. Mitchell, N.A. Duffie, J.A.and Duffie, T.L. Freeman, J.C. Mitchell, J.E. Braun, B.L. Evans, J.P. Kummer, R.E. Urban, A. Fiksel, J.W. Thornton, N.J. Blair, P.M. Williams, and D.E. Bradley. Trnsys, a transient system simulation program. Technical report, Solar Energy Lab, Univ. of Wisconsin-Madison, 2000.

[135] J. Knowles and J. Corne. On metrics for comparing non-dominated sets. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, volume 1, pages 711–716, 2002.

[136] Christoph Koch, Frank Cziesla, and George Tsatsaronis. Optimization of combined cycle power plants using evolutionary algorithms. *Chemical Engineering and Processing: Process Intensification*, 46(11):1151 – 1159, 2007. Special Issue on Process Optimization and Control in Chemical Engineering and Processing.

[137] John R. Koza. Genetic programming: a paradigm for genetically breeding populations of computer programs to solve problems. Technical report, Stanford, CA, USA, 1990.

[138] M. Krarti. An overview of artificial intelligence-based methods for building energy systems. *Journal of Solar Energy Engineering*, 125:331, August 2003.

[139] J.F. Kreider and J.S. Haberl. Predicting hourly building energy usage. *ASHRAE Journal*, 36(6), 1994.

[140] T. Krishnaiah, S. Rao, K. Madhumurthy, and K.S. Reddy. Neural network approach for modelling global solar radiation. *Journal of Applied Sciences Research*, (3):1105–1111, 2007.

[141] R. Kumar, S.C. Kaushik, and S.N. Garg. Heating and cooling potential of an earth-to-air heat exchanger using artificial neural network. *Renewable Energy*, (31):1139–1155, 2006.

[142] R. Kumar, A.R. Sinha, B.K. Singh, and U. Modhukalya. A design optimization tool of earth-to-air heat exchanger using a genetic algorithm. *Renewable Energy*, (33):2282–2288, 2008.

[143] Andrew Kusiak, Mingyang Li, and Zijun Zhang. A data-driven approach for steam load prediction in buildings. *Applied Energy*, 87(3):925 – 933, 2010.

[144] H.N. Lam. Intelligent computer control of air conditioning systems based on genetic algorithm and classifier system. In *Proc. Building Simulation 1995 Conf*, pages 151–157, 1995.

[145] J. Lam, K. Wan, and L. Yang. Solar radiation modelling using anns for different climates in china. *Energy Conversion and Management*, 49(5):1080–1090, May 2008.

[146] Greg Ward Larson and Rob Shakespeare. *Rendering With Radiance: The Art And Science Of Lighting Visualization*. Booksurge Llc, 2004.

[147] P. Lauret, H. Boyer, C. Riviere, and A. Bastide. A genetic algorithm applied to the validation of building thermal models. *Energy and Buildings*, (37):858–866, 2005.

[148] Yann LeCun, Leon Bottou, Genevieve Orr, and Klaus Müller. Efficient backprop. In Genevieve Orr and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*, pages 546–546. Springer Berlin, 1998.

[149] K. Y. Lee and M.A. El-Sharkawi, editors. *Moder heuristic optimization techniques: theory and applications to power systems*. IEEE Press Series on Power Engineering. Wiley-IEEE, 2007.

[150] W-S. Lee and L-C. Lin. Optimal chiller loading by particle swarm algorithm for reducing energy consumption. *Applied Thermal Engineering*, (29):1730–1734, 2009.

[151] K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Applied Math.*, 2:164–168, 1944.

[152] Kangji Li and Hongye Su. Forecasting building energy consumption with hybrid genetic algorithm-hierarchical adaptive network-based fuzzy inference system. *Energy and Buildings*, In Press, Accepted Manuscript:–, 2010.

[153] Qiong Li, Qinglin Meng, Jiejin Cai, Hiroshi Yoshino, and Akashi Mochida. Predicting hourly cooling load in the building: A comparison of support vector machine and different artificial neural networks. *Energy Conversion and Management*, 50(1):90 – 96, 2009.

[154] Jian Liang and Ruxu Du. Thermal comfort control based on neural network for hvac application. In *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, pages 819 –824, 28-31 2005.

[155] E. Liitiäinen and A. Lendasse. ariable scaling for time series prediction: Application to the estsp'07 and the nn3 forecasting competitions. In *IJCNN 2007, International Joint Conference on Neural Networks*, pages 2812–2816, 2007.

[156] Y. Liu and X. Yao. Ensemble learning via negative correlation. *Neural Networks*, 12(10):1399–1404, 1999.

[157] A. López and L. Sánchez. An evolutionary algorithm for the off-line data driven generation of fuzzy controllers for intelligent buildings. In *Systems, man and cybernetics, 2004 IEEE international conference*, volume 1, pages 42–47, 2004.

[158] L. Lu, W. Cai, L. Xie, S. Li, and Y.C. Soh. Hvac system optimization—in-building section. *Energy and Buildings*, (37):11–22, 2005.

[159] T. Lu and M. Vijanen. Prediction of indoor temperature and relative humidity using neural network models: model comparison. *Neural Computation & Applications*, (18):345–357, 2009.

[160] L. Magnier and F. Haghighat. Multiobjective optimization of building design using trnsys simulations, genetic algorithm, and artificial neural network. *Building and Environment*, (45):739–746, 2010.

[161] Imran Maqsood, Muhammad Riaz Khan, and Ajith Abraham. An ensemble of neural networks for weather forecasting. *Neural Computing & Applications*, 13:112–122, 2004. 10.1007/s00521-004-0413-4.

[162] MATLAB. *version 7.10.0 (R2010a)*. The Mathworks Inc., Natick, Massachusetts, 2010.

[163] H. Matsumoto, Y. Ohsawa, S. Takahasi, T. Akiyama, H. Hanaoka, and O. Ishiguro. Startup optimization of a combined cycle power plant based on cooperative fuzzy reasoning and a neural network. *Energy Conversion, IEEE Transactions on*, 12(1):51 –59, mar 1997.

[164] W.S. McCulloch and W. H. Pitts. A logical calculus of the ideas a logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.*, (5):115–133, 1943.

[165] A. Mechaqrane and M. Zouak. A comparison of linear and neural network arx models applied a comparison of linear and neural network arx models applied to a prediction of the indoor temperature of a building. *Neural Computation & Applications*, (13):32–37, 2004.

[166] M.R.G. Meireles, P.E.M. Almeida, and M.G. Simoes. A comprehensive review for industrial applicability of artificial neural networks. *Industrial Electronics, IEEE Transactions on*, 50(3):585 – 601, jun. 2003.

[167] 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 & Management*, (44):1525–1534, 2003.

[168] G. Mihalakakou, M. Santamouris, and A. Tsangrassoulis. On the energy consumption in residential buildings. *Energy and Buildings*, 34(7):727 – 736, 2002.

[169] L. Mitas and H. Mitasova. Spatial interpolation. In D.J. Maguire P. Longley, M.F. Goodchild and D.W. Rhind, editors, *Geographical Information Systems: Principles and Applications*, pages 481–492. Wiley, 1999.

[170] Jin Woo Moon and Jong-Jin Kim. Ann-based thermal control models for residential buildings. *Building and Environment*, 45(7):1612 – 1625, 2010.

[171] N. Morel, M. Bauer, M. El-Khoury, and J. Krauss. neurobat, a predictive and adaptive heating control system using artificial neural networks. *International Journal of Solar Energy*, 21(2):161 – 201, 2001.

[172] M. Mossolly, K. Ghali, and N. Ghaddar. Optimal control strategy for a multi-zone air conditioning system using a genetic algorithm. *Energy*, (34):58–66, 2009.

[173] M.C. Mozer. The neural network house: An environment that adapts to its inhabitants. In *Proceedings of the American Association for Artificial Intelligence Spring Symposium on Intelligent Environments*, pages 110–114. AAAI Press, 1998.

[174] J. Mubiru. Predicting total solar irradiation values using artificial neural networks. *Renewable Energy*, 33(10):2329–2332, October 2008.

[175] K. S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1(1):4–27, March 1990.

[176] N. Nassif, S. Kajl, and R. Sabourin. Optimization of hvac control system strategy using two-objective genetic algorithm. *International Journal of HVAC&R Research*, 11(3):459, july 2005.

[177] Nabil Nassif, Samir Moujaes, and Mohammed Zaheeruddin. Self-tuning dynamic models of hvac system components. *Energy and Buildings*, 40(9):1709 – 1720, 2008.

[178] Rahul L. Navale and Ron M. Nelson. Use of genetic algorithms to develop an adaptive fuzzy logic controller for a cooling coil. *Energy and Buildings*, 42(5):708 – 716, 2010.

[179] M. G. Negoita and B. Reusch. *Real World Applications of Computational Intelligence*. Studies in Fuzziness and Soft Computing. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[180] Alberto Hernandez Neto and Flavio Augusto Sanzovo Fiorelli. Comparison between detailed model simulation and artificial neural network for forecasting building energy consumption. *Energy and Buildings*, 40(12):2169 – 2176, 2008.

[181] D.H. Nguyen and B. Widrow. Neural networks for self-learning control systems. *Control Systems Magazine, IEEE*, 10(3):18 –23, April 1990.

[182] S.O.T. Ogaji, L. Marinai, S. Sampath, R. Singh, and S.D. Prober. Gas-turbine fault diagnostics: a fuzzy-logic approach. *Applied Energy*, 82(1):81 – 89, 2005.

[183] M. Ohlsson, C. Peterson, H. Pi, T. Rögnvaldsson, and B. Söderberg. Predicting system loads with artificial neural networks - methods and results from the great energy predictor shootout. In *1994 Annual Proceedings of the American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.*, 1994.

[184] R. Ooka and K. Komamura. Optimal design method for building energy systems using genetic algorithms. *Building and Environment*, (44):1538–1544, 2009.

[185] H.K. Ozturk, O.E. Canyurt, A. Hepbasli, and Z. Utlu. Residential-commercial energy input estimation based on genetic algorithm (ga) approaches: an application of turkey. *Energy and Buildings*, (36):175–183, 2004.

[186] R. Parameshwaran, R. Karunakaran, C. Vinu Raja Kumar, and S. Iniyan. Energy conservative building air conditioning system controlled and optimized using fuzzy-genetic algorithm. *Energy and Buildings*, 42(5):745 – 762, 2010.

[187] G.V. Pariswad, R.K. Bhardwaj, and V.K. Nema. Prediction of monthly-mean hourly relative humidity, ambient temperature, and wind velocity for india. *Renewable Energy*, 13(3):363–380, 1998.

[188] D.B. Parker. Learning logic. In *Invention Report*, pages 581–564. 1982.

[189] C.O. Pedersen, R.J. Liesen, R.K. Strand, D.E. Fisher, L. Dong, and P.G. Ellis. A toolkit for building load calculations. Technical report, ASHRAE, Atlanta, GA, 2000.

[190] Riccardo Poli. Analysis of the publications on the applications of particle swarm optimisation. *J. Artif. Evol. App.*, 2008:3:1–3:10, January 2008.

[191] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2010. ISBN 3-900051-07-0.

[192] I. Rechenberg. Cybernetic solution path of an experimental problem. Technical report, Ministery of Aviation, 1965.

[193] Colin R. Reeves. Genetic algorithms. In Michel Gendreau and Jean-Yves Potvin, editors, *Handbook of Metaheuristics*, volume 146 of *International Series in Operations Research & Management Science*, pages 109–139. Springer US, 2010.

[194] W. Remus and M. O'Connor. Neural networks for time-series forecasting. In J.S. Armstrong, editor, *Principles of Forecasting: A Handbook for Researchers and Practitioners*. Kluwer Academic Publishers, 2001.

[195] Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *IEEE INTERNATIONAL CONFERENCE ON NEURAL NETWORKS*, pages 586–591, 1993.

[196] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychol. Rev.*, (65):386–408, 1958.

[197] A.E. Ruano, E.M. Crispim, E.Z.E. Conceição, and M.M.J.R. Lúcio. Prediction of building's temperature using neural networks models. *Energy and Buildings*, (38):682–694, 2006.

[198] A. Rubaai and R. Kotaru. Online identification and control of a dc motor using learning adaptation of neural networks. *Industry Applications, IEEE Transactions on*, 36(3):935 –942, 2000.

[199] D. E. Rumelhart and J. L. McClelland. *Parallel distributed processing*. MIT Press, Cambridge, MA, USA, 1986.

[200] D. Ruta. Crosstrained ensemble of neural networks for robust time series prediction. In *NISIS*, Tenerife, Spain, 2006.

[201] J. D. Schaffer, D. Whitley, and L. J. Eshelman. Combinations of genetic algorithms and neural networks: a survey of the state of the art. In *International Workshop on Combinations of Genetic Algorithms and Neural Networks, 1992., COGANN-92*, pages 1–37, 1992.

[202] J.D Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100, 1985.

[203] J. Schott. *Fault tolerant design using single and multicriteria genetic algorithm optimization*. Masters thesis department of aeronautics and astronautics, Massachusetts Institute of Technology, 1995.

[204] Z. Şen. Fuzzy algorithm for estimation of solar irradiation from sunshine duration. *Solar Energy*, 63:39–49, 1998.

[205] J. Snyman. *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-based Algorithms*. Springer-Verlag New York Inc, Dordrecht, 2005.

[206] J. Soares. Modelling hourly diffuse solar-radiation in the city of são paulo using a neural-network technique. *Applied Energy*, 79(2):201–214, October 2004.

[207] S. Sodoudi. Estimation of temperature, precipitation and evaporation with neuro-fuzzy method. In *workshop of statistical downscaling*, Oslo, 2005.

[208] E.D. Sontag. *Recurrent Neural Networks: Some Systems-Theoretic Aspects*. PhD thesis, Dept. of Mathematics, Rutgers University, NB, Etats-Unis, 1996.

[209] F. Spinelli, A. Maccari, E. Cogliani, and M. Milone. La misura e la stima della radiazione solare: l'archivio dell'enea e il sito internet dell'atlante italiano della radiazione solare. *Energia e Innovazione*, (1), 2008.

[210] N. Srinivas and K Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[211] Lukas G. Swan and V. Ismet Ugursal. Modeling of end-use energy consumption in the residential sector: A review of modeling techniques. *Renewable and Sustainable Energy Reviews*, 13(8):1819 – 1835, 2009.

[212] M.S. Tanvir and I.M. Mujtaba. Neural network based correlations for estimating temperature elevation for seawater in msf desalination process. *Desalination*, (195):251–272, 2006.

[213] H. Tatli and Z. Şen. A new fuzzy modelling approach for predicting the maximum daily temperature from a time series. *Turkish Journal of Engineering and Environmental Sciences*, (23):173–180, 1999.

[214] F. Tay. Application of support vector machines in financial time series forecasting. *Omega*, 29(4):309–317, August 2001.

[215] F. Tetsuya. An optimum start up algorithm for combined cycle. *Transactions of the Japan Society of Mechanical Engineers*, 67(660):2129–2134, 2001.

[216] Thomas, Bertil, Soleimani-Mohseni, and Mohsen. Artificial neural network models for indoor temperature prediction: investigations in two buildings. *Neural Computing & Applications*, 16(1):81–89, January 2007.

[217] P. Thornton, S. Running, and M. White. Generating surfaces of daily meteorological variables over large regions of complex terrain. *Journal of Hydrology*, 190(3-4):214–251, March 1997.

[218] Stefan Uhlenbruck and Klaus Lucas. Exergoeconomically–aided evolution strategy applied to a combined cycle power plant. *International Journal of Thermal Sciences*, 43(3):289 – 296, 2004.

[219] Z. A. Vale, H. Morais, H. Khodr, Bruno Canizes, and Joao Soares. Technical and economic resources management in smart grids using heuristic optimization methods. pages 1–7, jul. 2010.

[220] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[221] S. Wang and X. Jin. Model-based optimal control of vav air-conditioning system using genetic algorithm. *Building and Environment*, (35):471–487, 2000.

[222] S. Wang and X. Xu. Parameter estimation of internal thermal mass of building dynamic models using genetic algorithm. *Energy Conversion and Management*, (47):1927–1941, 2006.

[223] Shengwei Wang and Youming Chen. Fault-tolerant control for outdoor ventilation air flow rate in buildings based on neural network. *Building and Environment*, 37(7):691 – 704, 2002.

[224] W. Wang, R. Zmeureanu, and H. Rivard. Applying multi-objective genetic algorithms in green building design optimization. *Building and Environment*, (40):1512–1525, 2005.

[225] Z. Wang, K. Tang, and X. Yao. Multi-objective approaches to optimal testing resource allocation in modular software systems. *IEEE Transactions on Reliability*, 59(3):563–575, September 2010.

[226] P.J. Werbos. *Beyond regression: new tools for prediction and analysis in the behavioral sciences*. PhD thesis, Harvard University, Cambridge, MA, 1974.

[227] Halbert White. *Artificial Neural Networks: Approximation and Learning Theory*. Blackwell Publishers, Inc., Cambridge, MA, USA, 1992.

[228] M.T. Wishart and R.G. Harley. Identification and control of induction machines using artificial neural networks. *Industry Applications, IEEE Transactions on*, 31(3):612 –619, 1995.

[229] S.L. Wong, K.K.W. Wan, and T.N.T. Lam. Artificial neural networks for energy analysis of office buildings with daylighting. *Applied Energy*, (87):551–557, 2010.

[230] J.A. Wright, H.A. Loosemore, and R. Farmani. Optimization of building thermal design and control by multi-criterion genetic algorithm. *Energy and Buildings*, (34):959–972, 2002.

[231] X. Xu and S. Wang. Optimal simplified thermal models of building envelope based on frequency domain regression using genetic algorithm. *Energy and Buildings*, 39(5):525 – 536, 2007.

[232] F. Yamada, K. Yonezawa, S. Sugawara, and N. Nishimura. Development of air-conditioning control algorithm for building energy saving. In *Control Applications, 1999. Proceedings of the 1999 IEEE International Conference on*, volume 2, pages 1579 –1584 vol. 2, 1999.

[233] J. Yang, H. Rivard, and R. Zmeureanu. On-line building energy prediction using adaptive artificial neural networks. *Energy and Buildings*, 37(12):1250 – 1259, 2005.

[234] J. Yi. *Building environment system simulation and analysis — DeST*. Building construction Press, 2006.

[235] L.A. Zadeh. Fuzzy sets. *Information Control*, 8:338–353, 1965.

[236] Lotfi A. Zadeh. Possibility theory and soft data analysis. *Mathematical Frontiers of the Social and Policy Sciences*, pages 69–129, 1981.

[237] G.P. Zhang and V.L. Berardi. Time series forecasting with neural network ensembles: An application for exchange rate prediction. *Journal of the Operational Research Society*, 52(6):652, 2001.

[238] Guoqiang Zhang, B. Eddy Patuwo, and Michael Y. Hu. Forecasting with artificial neural networks:: The state of the art. *International Journal of Forecasting*, 14(1):35–62, 1998.

[239] L. Zhang, N. Zhang, F. Zhao, and Y. Chen. A genetic-algorithm-based experimental technique for determining heat transfer coefficient of exterior wall surface. *Applied Thermal Engineering*, (24):339–349, 2004.

[240] L. Zhou and F. Haghighat. Optimization of ventilation system design and operation in office environment, part i: Methodology. *Building and Environment*, (44):651–656, 2009.

[241] L. Zhou and F. Haghighat. Optimization of ventilation systems in office environment, part ii: Results and discussions. *Building and Environment*, 44(4):657 – 665, 2009.

[242] J. Zhu. *Optimization of Power Systems Operation*. IEEE Press Series on Power Engineering. Wiley-IEEE, 2009.

[243] H.-J. Zimmermann. *Fuzzy set theory-and its applications*. Kluwer Academic Publishers, Norwell, MA, USA, 3rd ed. edition, 1996.

[244] E. Znouda, N. Ghrab-Marcos, and A. Hadj-Alouane. Optimization of mediterranean building design using genetic algorithms. *Energy and Buildings*, (39):148–153, 2007.