Finding Representative Scenarios in Wind Power or Load Forecast

Tiago André da Mota Jesus Rodrigues

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Pr. Dr. Vladimiro Henrique Barrosa Pinto de Miranda
Second Supervisor: Dr. Jean Sumaili

July 24, 2013
A Dissertação intitulada

"Finding Representative Scenarios in Wind Power or Load Forecast"

foi aprovada em provas realizadas em 16-07-2013

o júri

Presidente Professor Doutor João Abel Peças Lopes
Professor Catedrático do Departamento de Engenharia Eletrotécnica e de
Computadores da Faculdade de Engenharia da Universidade do Porto

Professor Doutor José Boaventura Ribeiro da Cunha
Professor Associado c/ Agregação do Departamento de Engenharia Eletrotécnica e
de Computadores da Universidade de Trás os Montes e Alto Douro

Professor Doutor Vladimiro Henrique Barrosa Pinto de Miranda
Professor Catedrático do Departamento de Engenharia Eletrotécnica e de
Computadores da Faculdade de Engenharia da Universidade do Porto

Doutor Jean Sumaili
Investigador do INESC - TEC

O autor declara que a presente dissertação (ou relatório de projeto) é da sua
exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente
autorizado. Os resultados, ideias, parágrafos, ou outros extratos tomados de ou
inspirados em trabalhos de outros autores, e demais referências bibliográficas
usadas, são corretamente citados.

Tiago André da Mota Jesus Rodrigues
Autor - Tiago André da Mota Jesus Rodrigues

Faculdade de Engenharia da Universidade do Porto
Abstract

The increasing introduction of energy production means through wind generators causes a greater variability in an electric energy system due to the uncertainty and intermittency of the wind. The need to work with a large number of scenarios to accurately represent the uncertainty of a wind power or load forecast has hindered the use of stochastic models to solve unit commitment problems due to the resulting heavy computational burden.

To answer this problem, scenario reduction techniques were proposed that are capable of sufficiently reducing the number of scenarios and still maintaining a good representation of the uncertainty of a forecast. However, several works neglect the danger that a scenario reduction implies, certain scenarios that have low representation are not properly accounted. These scenarios may have significant consequences in case they occur.

This work seeks to resolve the problem of a faithful representation of interesting scenarios with low occurrence in a set of forecast scenarios through a clustering technique. This clustering technique, which is based in an optimization of Entropy criteria, finds, around a fixed number of representative scenarios, the scenarios that best cluster the set in respect to similarity measures and maximizing an objective function.

To solve the optimization problem the metaheuristics Simulated Annealing and Evolutionary Programming were used. A comparison between the metaheuristics and the similarity measures was done to verify which method best performs and to assess the influence the similarity measure has in clustering the scenarios.

With the correct analysis, the proposed methodology successfully solves the problem of identifying the relevant low represented scenarios and produces a reduced set of scenarios for use in stochastic programming in unit commitment problems.
Resumo

A crescente introdução de meios de produção de energia através de geradores eólicos leva a uma maior variabilidade de um sistema eléctrico. A necessidade de se trabalhar com um elevado número de cenários para caracterizar a incerteza de uma previsão de produção eólica ou de carga dificultou o uso de modelos estocásticos na resolução de problemas de escalonamento e despacho devido à resultante carga computacional elevada.

Em resposta a este problema surgiram técnicas de redução de cenários que são capazes de reduzir suficientemente o número de cenários mantendo uma boa representação da incerteza associada a uma previsão. No entanto, vários trabalhos ignoram o perigo que uma redução de cenários implica, certos cenários que têm pouca representação não são contabilizados devidamente. Estes cenários podem ter consequências graves caso ocorram.

Este trabalho pretende resolver o problema de uma fiel representação de cenários interessantes com pouca ocorrência num conjunto de cenários de previsão através de uma metodologia de agrupamento de cenários. A técnica de agrupamento assenta numa optimização de critérios de Entropia encontrando, a partir de um número fixo de cenários representativos, quais os cenários que melhor agrupam o conjunto de cenários, com base em métricas de semelhança, maximizando a função objetivo.

Para resolver o problema de optimização foram usadas as meta-heurísticas, Arrefecimento Simulado (Simulated Annealing) e Programação Evolucionária (Evolutionary Programming). Uma comparação às meta-heurísticas e às métricas de semelhança usadas foi feita para verificar qual dos métodos melhor se adequa ao problema e verificar a influência que a escolha da métrica de semelhança tem no agrupamento de cenários.

Com a correcta análise ao problema em questão, a metodologia proposta resolve com sucesso o problema de se identificar os cenários relevantes com pouca representação e produz um conjunto de cenários reduzido para ser usado em programação estocástica para problemas de escalonamento e despacho.
Acknowledgements

Firstly, I would like to thank my Supervisors at INESC TEC, Pr. Vladimiro Miranda and Dr. Jean Sumaili, for their unwavering support, motivation, guidance and expertise that helped the completion this work and their contributions to my graduation and my identity as a future Electrical Engineer. I would also like to express my gratitude to Hrvoje Keko for his support and ideas for this work.

My sincere gratitude towards my teachers who helped in my graduation, who taught me more than knowledge about this course and whose expertise and their way of being in life I will always admire.

I would also like to thank my colleagues, who as me are also finishing this course at INESC TEC. My special thanks go to my great friends Afonso Teixeira, Carlos Guerreiro, João Sousa, Jorge Carvalho, Ricardo Calçarão and Tiago Pinto for their support, friendship and the great moments we shared in the past 5 years at FEUP.

My special thanks to my father and mother for raising and educating me, my brother and his girlfriend for their ever present companionship and support.

There are no amount of words that I could use to thank the person who supported me the most and gave me the strength and confidence to face this last challenge in my graduation, my love Joana Vasconcelos.

Finally I thank every person who helped shape me into the person I am now, all of whom are still with me, those who unfortunately are not and those whose work and philosophies influenced me the most.

Tiago Rodrigues
“Building the future and keeping the past alive are one and the same thing.”

Hideo Kojima
# Contents

1 Introduction .................................................. 1
   1.1 Motivation - The need for Scenario Reduction ............... 1
   1.2 Objectives of this Dissertation .......................... 2
   1.3 Structure of the Dissertation .............................. 2

2 State of the Art .............................................. 5
   2.1 Forecasting .............................................. 5
      2.1.1 Load Forecasting ................................... 5
      2.1.2 Wind Forecasting .................................. 6
   2.2 Scenario Tree Reduction Techniques ....................... 7
      2.2.1 Scenario Trees ...................................... 7
      2.2.2 Scenario Tree Reduction Problem .................... 8
      2.2.3 Variations of Scenario Tree Reductions ............. 9
   2.3 Clustering Techniques ................................... 12
   2.4 Conclusions .............................................. 13

3 Reducing Scenarios - A new approach ......................... 15
   3.1 Probability Density Function Estimation .................... 15
   3.2 Information Theory ...................................... 16
      3.2.1 The Concept of Entropy ............................ 16
      3.2.2 Cluster Evaluation Functions ....................... 17
      3.2.3 Information Theoretic Mean Shift algorithm ........... 18
   3.3 Similarity Measures ..................................... 22
      3.3.1 Gaussian Kernel Similarity ......................... 22
      3.3.2 Cosine Similarity .................................. 22
      3.3.3 Kohonen Similarity ................................. 23
   3.4 Metaheuristics .......................................... 23
      3.4.1 Simulated Annealing ............................... 24
      3.4.2 Evolutionary Programming ......................... 25
   3.5 A New Clustering Technique ................................ 27
      3.5.1 Scenario and Cluster definitions .................. 27
      3.5.2 Objective Function ................................. 28
      3.5.3 Optimization Algorithms ........................... 28
      3.5.4 Choosing the Representative Scenario ............... 33

4 Results in Study Cases .................................... 35
   4.1 Used Datasets ............................................ 35
   4.2 Study Case 1: PDF estimation using Parzen Windows technique .... 38
4.3 Study Case 2: ITMS algorithm ................................................. 42
  4.3.1 Testing with Datasets 1, 2 and 3 ................................. 42
  4.3.2 Mode finding in wind power forecast scenarios ............. 46
4.4 Study Case 3: Scenario Reduction ............................. 49
  4.4.1 Performance Tests .................................................. 50
  4.4.2 Effects of the weight $\lambda$ ...................................... 52
  4.4.3 Scenario Reduction ............................................... 54
  4.4.4 Different Similarity Measures ................................. 59

5 Conclusions and Future Work ................................. 67
  5.1 Conclusions .......................................................... 67
  5.2 Future Work .......................................................... 68

A ITMS Results .......................................................... 71
  A.1 Dataset 1 ............................................................ 71
  A.2 Dataset 2 ............................................................ 73
  A.3 Dataset 3 ............................................................ 75

B Article for submission ................................................. 79

References ............................................................. 83
## List of Figures

2.1 Scenario tree with 5 scenarios and 10 nodes [1] .................................................. 7
2.2 Load scenario tree for one week [2] ................................................................. 8
2.3 Illustration of CVaR and Var [3] ................................................................. 11
3.1 Estimated probability density function for a double crescent shaped data set ... 16
3.2 Different classifications of metaheuristics [4] .................................................. 24
3.3 A cumulative distribution function ............................................................... 29
4.1 Double Crescent shape curves - Dataset 1 .................................................. 35
4.2 Random Gaussian Clusters - Dataset 2 ....................................................... 36
4.3 Random Gaussian Clusters with wider dispersion - Dataset 3 ...................... 36
4.4 Wind Power forecast scenarios for 3 days .................................................. 37
4.5 Wind Power forecast scenarios for 1 day .................................................. 37
4.6 Load forecast scenarios for 1 day ............................................................... 37
4.7 Contour plot of the PDF for dataset1 ....................................................... 39
4.8 Surface plot of the PDF for dataset1 ....................................................... 39
4.9 Contour plot of the PDF for dataset2 ....................................................... 40
4.10 Surface plot of the PDF for dataset2 ....................................................... 40
4.11 Contour plot of the PDF for dataset3 ....................................................... 41
4.12 Surface plot of the PDF for dataset3 ....................................................... 41
4.13 Mode finding results (right) compared to the PDF of Dataset 1 (left) ........ 43
4.14 Mode finding results (right) compared to the PDF of Dataset 2 (left) ........ 44
4.15 Mode finding results (right) compared to the PDF of Dataset 3 (left) ........ 44
4.16 Poor GBMS mode finding results ............................................................ 45
4.17 Principal curve finding results ............................................................... 45
4.18 Results of increasing the value of $\lambda$ .................................................. 46
4.19 Results of decreasing the value of $\sigma$ .................................................... 46
4.20 The modes of the dataset with wind power forecast scenarios .............. 47
4.21 Day 1 wind power forecast scenarios and its mode – 301 scenarios ...... 48
4.22 Day 2 wind power forecast scenarios and its mode – 300 scenarios ...... 48
4.23 Day 3 wind power forecast scenarios and its mode - 299 scenarios ...... 49
4.24 Objective Function behaviour during performance test with SA for wind scenarios 51
4.25 Objective Function behaviour during performance test with EP for wind scenarios 51
4.26 Objective Function behaviour during performance test with SA for load scenarios 51
4.27 Objective Function behaviour during performance test with EP for load scenarios 51
4.28 Objective Function behaviour during the load scenario reduction ........... 55
4.29 Identified load representative scenarios .................................................. 55
4.30 Clusters and representative scenario - clusters 1 to 16 ......................... 56
4.31 Clusters and representative scenario - clusters 17 to 25 ............................. 56
4.32 Objective Function behaviour during the wind scenario reduction .................. 57
4.33 Identified wind power representative scenarios ............................................ 58
4.34 Clusters and representative scenario - clusters 1 to 16 ............................... 58
4.35 Clusters and representative scenario - clusters 17 to 25 ............................. 59
4.36 Objective Function behaviour with Cosine similarity .................................. 60
4.37 Identified load representative scenarios with Cosine similarity .................... 61
4.38 Clusters and representative scenario with Cosine similarity - clusters 1 to 16 .... 61
4.39 Clusters and representative scenario with Cosine similarity - clusters 17 to 25 .... 62
4.40 Objective Function behaviour with Kohonen similarity .............................. 63
4.41 Identified load representative scenarios with Kohonen similarity .................. 63
4.42 Clusters and representative scenario with Kohonen similarity - clusters 1 to 16 .... 64
4.43 Clusters and representative scenario with Kohonen similarity - clusters 17 to 25 .... 64
A.1 Dataset 1 - Test 1 results ................................................................. 71
A.2 Dataset 1 - Test 2 results ................................................................. 71
A.3 Dataset 1 - Test 3 results ................................................................. 71
A.4 Dataset 1 - Test 4 results ................................................................. 71
A.5 Dataset 1 - Test 5 results ................................................................. 72
A.6 Dataset 1 - Test 6 results ................................................................. 72
A.7 Dataset 1 - Test 7 results ................................................................. 72
A.8 Dataset 1 - Test 8 results ................................................................. 72
A.9 Dataset 1 - Test 9 results ................................................................. 72
A.10 Dataset 1 - Test 10 results ............................................................. 72
A.11 Dataset 1 - Test 11 results .............................................................. 73
A.12 Dataset 1 - Test 2 results ............................................................... 73
A.13 Dataset 2 - Test 1 results ............................................................... 73
A.14 Dataset 2 - Test 2 results ............................................................... 73
A.15 Dataset 2 - Test 3 results ............................................................... 73
A.16 Dataset 2 - Test 4 results ............................................................... 73
A.17 Dataset 2 - Test 5 results ............................................................... 74
A.18 Dataset 2 - Test 6 results ............................................................... 74
A.19 Dataset 2 - Test 7 results ............................................................... 74
A.20 Dataset 2 - Test 8 results ............................................................... 74
A.21 Dataset 2 - Test 9 results ............................................................... 74
A.22 Dataset 2 - Test 10 results .............................................................. 74
A.23 Dataset 2 - Test 11 results .............................................................. 75
A.24 Dataset 2 - Test 12 results .............................................................. 75
A.25 Dataset 3 - Test 1 results ............................................................... 75
A.26 Dataset 3 - Test 3 results ............................................................... 75
A.27 Dataset 3 - Test 4 results ............................................................... 75
A.28 Dataset 3 - Test 5 results ............................................................... 75
A.29 Dataset 3 - Test 6 results ............................................................... 76
A.30 Dataset 3 - Test 7 results ............................................................... 76
A.31 Dataset 3 - Test 8 results ............................................................... 76
A.32 Dataset 3 - Test 9 results ............................................................... 76
A.33 Dataset 3 - Test 10 results ............................................................. 76
A.34 Dataset 3 - Test 11 results ............................................................. 76
A.35 Dataset 3 - Test 12 results ............................................................. 77
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Testing conditions and number of iterations for each Dataset</td>
<td>43</td>
</tr>
<tr>
<td>4.2</td>
<td>Final results of the mode finding to each dataset.</td>
<td>43</td>
</tr>
<tr>
<td>4.3</td>
<td>The conditions for the tests conducted using SA</td>
<td>50</td>
</tr>
<tr>
<td>4.4</td>
<td>The conditions for the tests conducted using EP</td>
<td>50</td>
</tr>
<tr>
<td>4.5</td>
<td>Performance Tests Results - SA</td>
<td>50</td>
</tr>
<tr>
<td>4.6</td>
<td>Performance Tests Results - EP</td>
<td>51</td>
</tr>
<tr>
<td>4.7</td>
<td>Conditions to evaluate the effects of varying the weight $\lambda$</td>
<td>52</td>
</tr>
<tr>
<td>4.8</td>
<td>Results for values of the weight $\lambda$ between 0 and 0.3</td>
<td>53</td>
</tr>
<tr>
<td>4.9</td>
<td>Results for values of the weight $\lambda$ between 0.4 and 0.7</td>
<td>53</td>
</tr>
<tr>
<td>4.10</td>
<td>Results for values of the weight $\lambda$ between 0.8 and 1.0</td>
<td>53</td>
</tr>
<tr>
<td>4.11</td>
<td>Results for values of the weight $\lambda$ between 0.985 and 0.995</td>
<td>54</td>
</tr>
<tr>
<td>4.12</td>
<td>Results for reduction from 193 to 25 load forecast scenarios</td>
<td>54</td>
</tr>
<tr>
<td>4.13</td>
<td>Results for reduction from 300 to 25 wind forecast scenarios</td>
<td>57</td>
</tr>
<tr>
<td>4.14</td>
<td>Results obtained using Cosine similarity in load forecast scenario reduction</td>
<td>60</td>
</tr>
<tr>
<td>4.15</td>
<td>Results obtained using Kohonen similarity in load forecast scenario reduction</td>
<td>62</td>
</tr>
</tbody>
</table>
List of Acronyms and Abbreviations

FEUP Faculdade de Engenharia da Universidade do Porto
INESC TEC Instituto de Engenharia de Sistemas e Computadores Tecnologia e Ciência
ISO Independent System Operator
TSO Transmission System Operator
DSO Distribution System Operator
CVaR Conditional Value at Risk
VaR Value at Risk
FSWC Forward Selection in Wait-and-see Clusters
ITMS Information Theoretic Mean Shift
AC Ant Colonization
TS Tabu Search
EC Evolutionary Computing
ES Evolution Strategies
GA Genetic Algorithms
EPSO Evolutionary Particle Swarm Optimization
EP Evolutionary Programming
SA Simulated Annealing
PDF Probability Density Function
CEF Cluster Evaluation Function
CDF Cumulative Distribution Function
PD Probability Distribution
IC Iteration Counter
WSC Worst Solution Counter
STC Same Temperature Counter
Chapter 1

Introduction

The first chapter of this Dissertation presents the main motivation for this work, a brief overview of the unit commitment problem, the importance of stochastic programming models and mainly the need for scenario reduction. Also described in this chapter are the objectives of this work and the main structure of this Dissertation. The work was developed at INESC TEC under the supervision and guidance of Professor Vladimiro Miranda and Dr. Jean Sumaili.

1.1 Motivation - The need for Scenario Reduction

In power systems, the unit commitment problem is defined as the set of decisions and the scheduling of the generators that will work over a twenty-four hour time period. It usually defines, given a known or forecasted load, the generators that will work over the time period, defining the ones that need to be turned on or turned off. The main objective in unit commitment is to minimize these operations’ costs respecting a set of constraints such as the generation meeting the load demand and the losses, upper and lower requirements of generators, minimum up/down time limits, generator turn on/off times and costs, technical transmission constraints and the spinning reserve. [5]

Given the rising concerns with the environment, new ways of clean energy production have been introduced, most notably Wind Power Generation in last few decades. Given the increasing penetration of this technology, decision makers need a reliable way to cope with the uncertainty and the intermittency of wind power generators. To address this problem, the multi-stage stochastic programming was introduced. Using these tools, one assumes that the uncertainty evolves as a discrete time continuous stochastic process, which can in turn, be described as an approximation of a finite probability distribution within the formulation of the unit commitment problem.

The problem that arose with the representation of the uncertainty of a given wind power forecast was that one would require a high number of forecast scenarios to accurately represent this uncertainty. This causes a heavy computational burden in the stochastic programming, rendering the problem intractable. In this situation the need for a reduction of the number of scenarios appeared which prompted several studies in this area. The reduced set of scenarios should still be
able to represent the uncertainty present in the original set of forecast scenarios to use in stochastic programming models.

There are some problems that may appear during the process of scenario reduction, for example, the low representation scenarios, i.e. outliers, may be excluded from the original set. These outliers may yield relevant features and losing these scenarios could cause significant consequences should they occur. This Dissertation will feature a methodology that can reduce the number of scenarios but still represent the outliers.

The main motivation for this work is to give decision makers more insight on the consequences of working with a reduced set of scenarios. Although this problem arose from the representation of wind power forecast scenarios, the same principles can applied to load forecast.

1.2 Objectives of this Dissertation

In this Dissertation, a new clustering technique is proposed that efficiently reduces the number of scenarios of wind or load forecast while retaining a representation of strange and underrepresented scenarios, but potentially relevant. The main focus of this methodology is to construct a set of representative scenarios that still represent the probability density function of the original scenario set.

In order to reduce the number of scenarios through clustering and still keep the underrepresented scenarios, the methodology will consist of an optimization problem with the maximization of the Entropy between the formed clusters and the minimization of the entropy in each cluster. To solve the optimization problem, two meta-heuristics were used: Simulated Annealing and Evolutionary Programming.

An article will also be written detailing the work developed in this Dissertation and will also be annexed to this document.

1.3 Structure of the Dissertation

Understanding that the scenario reduction is not a new one, several works and papers were researched and reviewed to aid and to better understand the problem in question. Chapter 2 features the literature review of the more relevant Scenario Reduction techniques along with some concerns that were pointed by the authors.

After reviewing the literature, several theories and tools were studied to compose the proposed methodology in this Dissertation. Thus, Chapter 3 features the main concepts and also the tools that were analysed during this work. The Information Theory and optimization algorithms that were used are firstly presented, finishing with the explanation behind the workings of the new methodology.

Naturally this new approach to reducing scenarios needed testing in order to validate it. But first, a few study cases are presented with tests to learn the concepts behind probability density estimation and the use of mean shift algorithms in mode finding. The final study case shows the
1.3 Structure of the Dissertation

tests that were conducted with the scenario reduction procedures. Chapter 4 features the results of these study cases.

Lastly, the Dissertation is concluded with the main conclusions taken from the tests and from the overall developed work. Besides the conclusions Chapter 5 also features recommendations for future work and new research questions.

The Appendices of this document features plots that were taken from testing of the Information Theoretic Mean Shift algorithm. These plots served to better understand the concepts behind this algorithm. Also featured in Appendix B is the article containing a long abstract of the developed work for submission to a conference or to a journal.
Chapter 2

State of the Art

The present chapter features a literature review concerning the scenario reduction techniques already proposed. A small introduction to the load and wind forecasting will be presented. The following sections will focus on the techniques and its strengths and concerns pointed by other authors. The second section will focus on the techniques based in scenario tree construction and the third section will focus in the application of clustering techniques and the creation of representative scenarios. The last section will feature the conclusions taken from the reviewed techniques.

2.1 Forecasting

Forecasting is, as defined in the Business Dictionary, “a planning tool that helps management in its attempts to cope with the uncertainty of the future, relying mainly on data from the past and present and analysis of trends”. [6]

Given that the nature of this work concerns scenario reduction techniques, the following subsections provide a small introduction to load and wind power forecasting and will not detail the models and techniques used in forecasting.

2.1.1 Load Forecasting

Usually unit commitment problems are resolved for a time horizon of 3 to 7 days, so the accurate description of the load variation in this time horizon is required. [5] Having an accurate forecast for the load demand will help electric utility companies “make important decisions on purchasing and generating electric power, load switching, and infrastructure development. Load forecasts are extremely important for energy suppliers, ISOs, financial institutions, and other participants in electric energy generation, transmission, distribution, and markets”. [7]

Load forecasting is usually divided in three main groups: the short-term forecasts ranging from one hour to one week; the medium forecasts usually range from one week to a year; the long-term forecasts focus on periods longer than a year. Naturally the longer the forecasting time period is, the accuracy of the forecasting will be inferior. [7]
Load forecasting takes an increasing importance for utility companies in deregulated energy markets in which the supply and demand fluctuations and changes of weather conditions increase the energy prices by tenfold during peak hours. In this cases, the use of short-term load forecasting can prevent the occurrence of overloads by estimating load flows. This kind of decisions and its timely implementation helps improving the network reliability and reduce the occurrence of equipment failure and blackouts. [7]

Regarding the methods for load forecasting, statistical techniques or artificial intelligence approaches like regression, neural networks or fuzzy logic can be used. Methods such as end-use and econometric approach are used for medium and long-term forecasting. For short-term forecasting, regression models, time series, neural networks or fuzzy logic can be used. [7]

2.1.2 Wind Forecasting

Although a clean type of energy production, power generation using the wind raises problems and concerns for power management. The variability and uncontrollable nature of the wind means that this kind of energy generation must require a different type of management unlike thermal power units in which the energy produced can be easily controlled.

To cope with the uncertainty of the wind, one requires accurate forecast models. These models are needed to ensure the security of energy supply, to help solve problems of congestion management, to plan secondary and tertiary reserves affected by wind power generation, to plan the maintenance of wind farms and to plan the operations in energy markets, among others. [8]

Wind power forecasting is needed by wind farm promoters, companies that are in charge of the installation and maintenance of wind farms, TSOs, ISO and energy traders, among others. [8]

Like in load forecasting there are several types for wind power forecasting in a time sense: nowcasting, very short-term and short-term forecasting. Nowcasting is used for forecasts in the next seconds or minutes and they are used in control and management of wind farms. Very short-term forecasts regard a time horizon to the next three hours and are used for production systems management, primarily in isolated states. Short-term forecast can span up to the next seven days and is used for management and planning of installation and management of wind farms, production management and energy trading. [8]

Forecasting may also be classified geographically: nationwide, for farm clusters, for each wind farm and for each wind power generator. Nationwide forecasts are important for system operators, market operators and traders and the aggregation process can take different approaches such as farm aggregation, farm cluster aggregation or multi-point global forecast and the forecast’s performance improves with the geographical aggregation. Farm cluster forecasts makes use of the high correlation of wind speed between farms and a production history for the farm cluster is needed. Regarding the forecasting for each wind farm, there is a good trade-off between the performance of the forecast and its cost, in terms of information details and computational burden. As with forecasting for farm clusters, it needs a production history for each wind farm and it’s the preferred solution for promoters, managers and builders of wind farms. Lastly, forecasting for each wind power generator demands more details about weather forecast and the production
history of the generator. It models the obstacles’ effects in several directions and the unavailability for each generator. [8]

The forecasting models are divided in three groups: numerical forecast models, physical models and statistical models. The first ones describe the evolution of atmospherical variables that define the weather state and its frontier conditions. Physical models model the fluid interactions in the field basing in the physical laws. The last ones are models based in the use of historical information of past forecasts although a clean type of energy production, as in load forecast, neural networks and regression models may be used. [8]

2.2 Scenario Tree Reduction Techniques

2.2.1 Scenario Trees

A Scenario Tree is an abstract structure of forecast scenarios that represents how the uncertainty evolves over time. A simple scenario tree is represented in Figure 2.1. [1]

Figure 2.1: Scenario tree with 5 scenarios and 10 nodes [1]

A scenario is defined as the complete path between the initial node n1 and one of the leaves, for example n6. The tree in Figure 2.1 has got five scenarios. These trees are useful for “the formulation of multi period dynamic decision models as multi-stage stochastic programs”. A multi-stage stochastic program will determine the optimal path that contains the set of decisions for each node of the scenario tree, given the available information at that point. The optimal decisions should anticipate future events, given that there are several succeeding nodes. [1]

An example of a load scenario tree can be seen in Figure 2.2. There are several methods to generate scenario trees. A literature review of these methods is available in [1] in Chapter III as the objective of this subsection is only to explain what a scenario tree is.
2.2.2 Scenario Tree Reduction Problem

In [2, 9] Dupačová et al. introduce the problem of the optimal scenario reduction as follows: “Determine a scenario subset of prescribed cardinality and a probability measure Q based on this set that is the closest to the initial distribution in terms of a natural (or canonical) probability metrics.” The optimal scenario reduction consists in determining a probability distribution Q that best approximates the original probability distribution P, composed of the original scenario dataset, with respect to a given distance d of probability measures and is supported by a much smaller number of scenarios. The probability distances trades off scenario probabilities and distances of scenario values.

The Scenario Reduction procedure is done through the deletion of non-important scenarios. The new probability of a kept scenario is equal to them sum of its former probability and of all probabilities of deleted scenarios that are closest to it in respect to a distance metric.

In [2] it’s introduced two different heuristics for the optimal reduction of the original N scenarios to N-k scenarios: backward reduction and forward selection. The aim of these heuristics is to find the optimal number of scenarios to be deleted, k. The backward reduction algorithm determines the k scenarios to be deleted by a lower bound technique and the optimal deletion of a single scenario is repeated recursively until the prescribed number N-k scenarios is met. The forward selection algorithm selects the N-k scenarios that are kept by an upper bound technique and is best used when the number of preserved scenarios is small.

Their tests show that after a reduction of 50% of the scenario tree, the optimal reduced tree still has about 90% relative accuracy and that the reduced trees created using forward selection are slightly better than the ones produced by the backward reduction, but it requires a higher computation times.

In the note published by Heitsch and Römisch in [10], they extend the previous work relying
directly on Fortet-Mourier metrics instead of using upper bounds. They no longer use generalized distances for scenarios, instead using reduced distances, “which are distances in the finite-dimensional scenario space and represent infima of certain optimization problems”. In the note, the optimal scenario reduction heuristic forward selection method is updated to address this change. They state that although the new algorithm “does not lead to optimality in general, the performance evaluation of its implementation in [9] is very encouraging”.

Gröwe-Kuska, Heitsch and Römisch apply the concept of scenario reduction and scenario tree construction to the power management problems in [1]. In the context of this kind of problems the authors chose the Kantorovich distance of multivariate probability distributions to compare the probability distribution of the original set of scenarios and the probability distribution of the reduced set of scenarios. They present a scenario tree construction algorithm that recursively reduces the number of nodes of the fan of individual scenarios by joining scenarios according to a scenario reduction algorithm such as simultaneous backward reduction or fast forward selection. The details for these algorithms are encountered in the mentioned article.

To test the scenario reduction and scenario tree construction, the authors solve a portfolio management for a hydro-thermal power system problem using Lagrangian Relaxation algorithms. The aim is to determine the trading activities and the production decisions of the generation system so that the expected revenue is maximized. Two experiments were done: the first regarding uncertain electrical load and spot market price and the second one only deals with uncertain electrical load. The first experiment is conducted with 54 scenarios with identical probabilities to model the distribution of the bivariate stochastic process for an hourly discretized time horizon of one week in summer for a hydro-thermal subsystem comprising of 4 thermal generation units and two pumped storage units. Their results show the relative accuracy of working with a reduced set of scenarios in Figures 9 and 10 presented in the fourth chapter of [1]. The second experiment “was designed to test the performance of the link between the Lagrangian Relaxation algorithm and the scenario tree construction algorithm”. The portfolio management problem is now comprised of 25 thermal units and 7 pumped-storage hydro units. The initial tree contains 100 scenarios, and the results are shown in figures 11, 12 and 13 of [1]. They conclude that their methodologies are useful for power management problems, saying that the optimal value of an optimization model can be approximated using a small number of scenarios.

### 2.2.3 Variations of Scenario Tree Reductions

The present subsection intends to show a few works that are based in the methodologies presented by the authors of the works in the previous subsection. Applications of scenario reduction in stochastic unit commitment problems, electricity trading and generation expansion planning are also shown.

In [11], Koc and Gosh propose a new technique that performs better than the earlier works. They argue that the techniques that only approximate the underlying distributions without attentions to the cost functions may produce weaker approximations of the optimal value. The motivation for their work arises from the following observation: “the computational complexity of
multi-stage program is driven not by the size of the support of \( Q \) (i.e. the number of distinct sample paths in the support) but by the number of nodes in the scenario tree implied by \( Q \). Thus the problem now concerns that the optimal approximation question should be imposed through constraining the number of unique support points the scenario tree can have in each stage instead of limiting the total number of unique scenarios in the approximation \( Q \). They propose that given a limited computational budget implied by a limited overall number of nodes in a scenario tree, “one could in a straightforward manner of search for the best approximation possible by imposing a combination of individual stage node limits that satisfies this overall bound”. They state that this approach holds the potential of being much faster.

To evaluate the limitations of scenario tree reduction approaches they conducted 2 experiments. The details of these experiments are present in the third chapter of [11]. They conclude that “obtaining scenario reductions by minimizing distributional distance measures alone without heed to the actual objective function leads to slightly poorer estimation of the optimal solution”.

Razali and Hashim proposed the application of the backward reduction algorithm to minimize the number of wind scenarios in [12], in the same manner that Gröwe-Kuska et al. did in [1]. They proposed the application of the “recursive backward reduction algorithm based in the Kantorovich distance to cluster large number of wind power scenarios in order to come up with a set of reduced scenarios to represent day-ahead wind power production of a wind farm” in Malaysia.

The backward reduction algorithm they used was a variation of the one proposed in [1], the main difference being that the calculation of the Kantorovich distance is done in a different manner. The authors state that the algorithm successfully determines the reduced set and the new probabilities for each scenarios, proving its usefulness in assisting power generation. They also conclude that the algorithm can be further improved by setting a new stopping criterion and find the optimal number of final scenarios.

Another interesting application of scenario reduction was done by Pineda and Conejo for risk-averse electricity trading in [3]. These authors make use of the fast forward selection algorithm developed in [9]. As stated, decision makers are risk averse, so future market decisions are made basing on the expected profit but considering the possibility of experiencing low profits. They define their optimization problem to maximize the value of the Conditional Value at Risk (CVaR), a risk measure commonly used in portfolio optimization. “The CVaR of a probability distribution with a confidence level \( \alpha \) is defined as the mean value of these scenarios with lowest profits and with an accumulated probability equal to \( 1-\alpha \). A different measure, the Value at Risk (VaR) “is equal to the profit such that the probability for a profit being less than this value is equal to \( 1-\alpha \). In simpler terms, the CVaR is the mean value of the profits lower than the VaR. Figure 2.3 illustrates this.

The scenario reduction algorithm starts by merging all the scenarios with a profit higher than VaR into a single scenario with probability equal to \( \alpha \). Since only the scenarios with a profit lower than VaR are needed to calculate the CVaR, these are the ones that will be reduced using the fast forward selection algorithm. [3]

Pineda and Conejo also review the metrics used to compare the definition of the distance
between two scenarios in section 2.3 of the article. They compare their method, the distance between two scenarios is calculated using the absolute difference of an auxiliary variable for each scenario, to other two techniques: the one used in [1, 2, 9, 10], defined as the norm of difference of the vectors containing the values of the stochastic process for the two scenarios; and another that defines the distance between two scenarios as the absolute value of the difference between the objective function of the corresponding single-scenario optimization problems in [13]. The results of their tests conclude that the technique used in [1, 2, 9, 10] performs less well than the proposed techniques in [3] and [13]. They also state that the technique in [13] outperforms theirs when the risk aversion parameter is higher.

The authors conclude that their methodology is effective in reducing the number of scenarios for continuous and discrete stochastic processes. They also affirm that the reduced scenario sets results in decisions identical or very close to the ones pertaining to the original scenario set.

Another application of the scenario reduction concept in a risk based environment was done in [14]. The authors apply the scenario reduction concept to risk based generation expansion planning. As in [3], the main value to optimize is the CVaR. The proposed methodology is based in the forward selection algorithm previously mentioned, but to mitigate its computational complexity, they propose the use of a combined heuristic called Forward Selection in Wait-and-see Clusters (FSWC) to reduce a large scenario set.

The authors notice that the classical scenario reduction methods focus only on the scenario parameters, failing to “account for where the uncertainties appear in the mathematical formulation or their impacts on the solution”. The use of the FSWC heuristic to incorporate the influences of the first-stage decisions requires the solution for a deterministic “wait and see” for every scenario to obtain specific first-stage decision for each scenario. The forward selection heuristic is applied after grouping the first-stage decisions based on their similarities. The process is repeated as
necessary to meet the desired number of reduced scenarios. [14]

It is concluded that the revised method, FSWC, performs better than the classical forward selection algorithm: it is capable of detecting the differences of generation capacities of each type of generator, based on the wait-and-see solutions for each expansion strategy more clearly. They also note that the computational time is smaller when using FSWC instead of the classical forward selection.

2.3 Clustering Techniques

More recently new approaches to the scenario reduction problem have been developed that use clustering techniques for scenario reduction. These clustering techniques aim to group the similar scenarios into groups and represent this group by a single scenario, usually called representative scenario. The aim of these representative scenarios is to accurately represent the Probability Density Function (PDF) of the wind power forecast.

In [15], the authors present a clustering technique to reduce the number of scenarios in a wind power forecast generated under a Monte Carlo process. The unimodal structure of the scenarios was confirmed using the Information Theoretic Mean Shift (ITMS) algorithm. This algorithm will be detailed further in this Dissertation.

The proposed methodology assumes that a large scenario set is sampled from the PDF representing the historical (observed) error distribution using a Monte Carlo sampling process. The high density areas are then replaced by a representative or focal scenario. The clustering approach is used without assuming any particular temporal development characteristic of error. [15]

In this work, the authors used forecasted and realized wind data to produce the wind power scenarios. The intended approach organizes this large set of scenarios, based on a principle of maximum density, in small set of clusters with an empirical probability and represent each cluster with a focal scenario. [15]

To cluster the scenarios, the metric to define the distance between two scenarios was the maximum deviation. The reduction procedure is done by finding the scenario that has the highest number of neighbours according to a desired tolerance. Two scenarios are considered neighbours if the maximum deviation between them is inferior to the desired tolerance. This tolerance must be viewed as a trade-off between the scenario reduction capability and accuracy, the higher the value, the higher the scenario reduction capability and the accuracy is lower. After creating a cluster and identifying its members, it’s needed to find the focal scenario. This focal scenario may be the attracting scenario or the mean value of the cluster or even using the ITMS to find the mode of the cluster. The last two cases, however, produce artificial scenarios, meaning that they were not created in the original scenario creation process. Choosing the first option, may not yield better results as “the total expected value may deviate from the expected value of the whole set of scenarios”. The choice of the focal scenario depends on the purpose of the scenario reduction. The probability of the focal scenario is then calculated as the ratio between the number of members of its cluster and the total number of scenarios. [15]
2.4 Conclusions

Their tests show that a large set of wind power scenarios can effectively be reduced by a smaller set. The reduction capability depends on the desired tolerance, the distance metric used and the scenario dispersion. The choice of the tolerance and the distance metric must be done in accordance to the problem in question and the accuracy needed for the results.

The article in [16] also by the previous authors, features the same scenario reduction approach but uses an evolutionary optimization algorithm to cluster the scenarios, finding the areas of high density and replacing the clusters with its more representative scenario. The same metric as before is used and a tolerance for this must also be defined.

To find the area of maximum probability density, they use an Evolutionary Particle Swarm Optimization (EPSO) to find the pattern that is able to aggregate the maximum number of scenarios such that the maximum deviation between the pattern and the scenarios is inferior to the admitted tolerance. EPSO is an optimization algorithm that relies on a set of “moving” solutions. More details for EPSO are present in the paper [16].

The scenario reduction methodology begins by inputting a large set of scenarios and defining an admissible tolerance. Using EPSO, the scenario with the highest aggregation capability is found and the scenarios aggregated by this one are grouped in a cluster. The process is repeated while the maximum density area contains more than one scenario. The remaining scenarios are considered to be outliers. [16]

The tests conducted by the authors show that for a set of 1000 scenarios, 31 focal scenarios and 64 outliers have been identified that represent about 93.6% of the initial set, with the remaining percentage representing the outliers. They also point the usefulness of their methodology: risk can be quantified by saving the number of selected focal scenarios and have information on the risk level accepted of the accepted choice, for example, “the risk of not including a representative behaviour or pattern of the wind power series”. A validation problem was also posed, using a simplified stochastic unit commitment problem to validate the reduced scenario set. For comparison purposes the problem was solved using the full set of scenarios, the clustered reduced set mentioned earlier with two variations, choosing the focal scenario as one present in the cluster and the other variation being that the focal scenario is the average value of the cluster, and a set of 95 randomly chosen scenarios. The results clearly show that using clustering approach achieves better results and for this case, averaging the scenarios in a cluster to create its focal scenario achieves better results. The computational effort was also measured, concluding that reducing the scenarios by tenfold, yields a run time reduction of about 16 times. [16]

2.4 Conclusions

The papers presented in this State of the Art show that the problem of scenario reduction for stochastic programming is not a new one and several authors published their methods and debated about others. Of all of the analysed works, the ones developed by Dupačová et al in [1, 2, 9, 10] are the most cited ones. Their work was adapted and used several times in the other papers that some authors point out some concerns: Koc and Ghosh advise in [9] that some approximations
may provoke unintended errors in the optimal values; Razali and Hashim in [12] note that the forward selection method applied to a small set of scenarios is selected in the new set, but some probabilities will be neglected in some extreme cases; Pineda and Conejo in [3] also address the issues of the distances metrics used and Feng and Ryan in [14] also note that classical methods focus only on the scenario parameters, neglecting the impacts of first-stage decisions.

Keko et al propose the use of clustering approaches to create a reduced set of scenarios. They too address the problem of choosing an appropriate measure to cluster the scenarios and choosing the representative scenario for each cluster. Of all the analysed works, [15] and [16] have more importance for this work as the proposed methodology is also a clustering-based approach. Also the forecasting techniques mentioned early in the chapter provide small insight to the problem of load and wind power forecasting, but they are not the main concern of this work.
Chapter 3

Reducing Scenarios - A new approach

This chapter will focus on the proposed methodology and the concepts that are used in this work. Some tools and methods that were studied during this work but were not used in the final version of the proposed methodology like a PDF estimator using Parzen windows or the ITMS algorithm will be detailed and shown examples of applications. Section 3.1 provides an introduction to the problem of PDF estimation, section 3.2 features the Information Theoretic concepts that were used in the development of this work, section 3.3 will focus on the similarity measures analysed in this work, section 3.4 explains the meta-heuristics used to solve the optimization problem and section 3.5 details the proposed methodology and will feature the main algorithms.

3.1 Probability Density Function Estimation

Parzen Window Density Estimation is a non-parametric density estimation technique invented by Emanuel Parzen in early 1960. In a parametric technique one has to assume that the form of the PDF is known. However, many physical processes are not explained using simple parametric models. As such, one would need a more appropriate technique that does not make any assumptions about the PDFs. Non-parametric estimation techniques achieve this, assuming only that the PDFs are smooth functions and can represent arbitrary PDFs given sufficient data. [17]

The Parzen Window Density Estimation assumes that given an instance of the random sample, \(x\), the procedure estimates the PDF \(P(x)\) from which the sample was derived. It overlaps kernel functions placed at each element of a sample, calculating the contribution that \(x_i\) has to the overall PDF estimate. [18]

This technique assumes that one can estimate the value of the PDF at any point, for example, \(y\). By placing a window function at this point, one can determine the contribution of the observation \(y_i\) to this window. The PDF value would be determined by the sum of the total combinations to this window. [18]

Considering a data set \(X(i) = \{x_1(1), \ldots x_i(n)\}^T | i = 1 \ldots N\), the probability density estimate can
be given by:

\[ P(X) = \frac{1}{N} \sum_{i=1}^{N} G\left( \left\| \frac{x-x_i}{\sigma} \right\|^2 \right) \]  \hspace{1cm} (3.1)

The kernel function \( G \) should be a Gaussian kernel with bandwidth \( \sigma > 0 \) as in Equation 3.2:

\[ G(t) = e^{-\frac{t^2}{2}} \]  \hspace{1cm} (3.2)

As an example Figure 3.1, shows the estimated PDF for a double crescent shaped with random distribution data set.

![Figure 3.1: Estimated probability density function for a double crescent shaped data set](image)

3.2 Information Theory

Information Theory is, according to the FreeDictionary.com, “a branch of mathematics that mathematically defines and analyses the concept of information. Information theory involves statistics and probability theory, and applications include the design of systems that have to do with data transmission, encryption, compression, and other information processing”. [19]

In this section, the concepts of Information Theory that were used in this work will be explained. The notion of entropy introduced by Renyi in [20] and its applications are the main basis for this work.

3.2.1 The Concept of Entropy

The notion of entropy was first introduced by Shannon in [21] and measures the uncertainty about a stochastic event, i.e., it quantifies the measure of information present in this event. [22] Renyi proposed in [20] a new way to estimate the measure of information in a Probability Distribution (PD) that later became known as Renyi’s Entropy and can be expressed in Equation 3.3. and
provided the starting point for an easier non-parametric estimator for entropy.

\[ H_R(x) = \frac{1}{1-\alpha} \ln \int f^\alpha(x) dx \quad \alpha > 0 \quad \alpha \neq 1 \]  

(3.3)

The authors in [22] propose the use of a Parzen Window Method to estimate the PDF to use Renyi’s Entropy described in Equation 3.3. They used the Gaussian kernel, \( G \), present in Equation 3.4:

\[ G(x) = \frac{1}{\sqrt{(2\pi)^n det(\Sigma_x)}} \times \exp \left( -\frac{1}{2} (x - x_i)^T \Sigma_x^{-1} (x - x_i) \right) \]  

(3.4)

Where \( \Sigma_x \) is the \( T \times T \) diagonal covariance matrix for datasets in each element has a dimension \( T \). Equations 3.5 and 3.6 show how one can calculate the Renyi’s Entropy without the need of any numerical evaluation of integrals using the notion that the authors call "Renyi’s Quadratic Entropy" for a data set with \( N \) elements and \( p(x) \) as its PDF:

\[ H_R(x) = -\ln \int p^2(x) dx \quad \alpha = 2 \]  

(3.5)

\[ H_R(x) = -\ln \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} G(x_i - x_j, \Sigma_x) \]  

(3.6)

They also call the quantity in Equation 3.7, Information Potential "since it is a positive decreasing function of the distance between samples \( x_i \) and \( x_j \), similar to the potential energy between physical particles".

\[ V(X) = \sum_{i=1}^{N} \sum_{j=1}^{N} G(x_i - x_j, \Sigma_x) \]  

(3.7)

### 3.2.2 Cluster Evaluation Functions

Principe et al propose several Cluster Evaluation Functions (CEF) in [22], but only two were needed for this work: Renyi’s Entropy for a single cluster and the Entropy between multiple clusters. The procedure to calculate the Entropy for a cluster is described in the previous subsection and only the procedure to calculate the Entropy between multiple clusters will be detailed in this section.

Principe and Gokcay first introduced a generalization for the CEF to more than two clusters in [22]. They stated that the main objective for this CEF was to measure the divergence between different clusters. Jenssen et al proposed in [23] the use of this between-cluster entropy in their new clustering algorithm.

Considering a data set with \( K \) clusters \( C_1, \ldots, C_K \) with \( N_1, \ldots, N_K \) elements that \( N_1 + \ldots + N_K = M \), the number of elements of the whole set of data, and defining a membership function, \( M(x_i, x_j) \) that equals one when if \( x_i \) and \( x_j \) belong to different clusters and zero if not, the expression of this
Reducing Scenarios - A new approach

Entropy is given by,

\[ H(C_1, \ldots, C_K) = -\ln V(C_1, \ldots, C_K) \]

\[ V(C_1, \ldots, C_K) = \frac{1}{2\prod_{k=1}^{K} N_k} \sum_{i=1}^{M} \sum_{j=1}^{M} G(x_i - x_j, \Sigma_x) \] (3.8)

One should note that if the clusters are well separated, \( V \) will exhibit a small value and, consequently, \( H \) will have a large value. From this, one could understand that Entropy between clusters can be used as a tool to measure the diversity between clusters, having a higher value if the clusters are less similar.

In [24] Sumaili and Chicco apply this concept to the classification of daily electrical patterns and introduce non-Euclidean metrics to measure the similarity between load scenarios and clusters by exploiting the entropy theory introduced by Shannon and Renyi. These similarity metrics will be discussed further in this chapter.

3.2.3 Information Theoretic Mean Shift algorithm

The Information Theoretic Mean Shift Algorithm (ITMS) is a cost function introduced by Rao et al in [25, 26] that aims to capture the “pre dominant structure” in the data. The cost function minimizes the entropy of the data subject to the constraint that the Cauchy-Schwarz distance between the new and the original dataset is fixed to some constant value. This generalized algorithm also addresses the already proposed Gaussian Mean Shift (GMS) and the Gaussian Blurring Mean Shift (GBMS) as special cases. ITMS is also capable of retrieving the principal curve in the data.

Considering the dataset \( X_0 = (x_i)_{i=1}^{N} \in \mathbb{R}^D \), the PDF can be estimated by a non-parametric method such as Parzen Window Technique detailed in Section 3.1 and Equation 3.1, using a simplified Gaussian Kernel such as in Equation 3.2. To find the modes of the PDF, one can solve the stationary point equation \( \nabla p(x) = 0 \) through an iterative stationary point scheme \( x^{(\tau+1)} = m(x^{\tau}) \) in which:

\[ m(x) = \frac{\Sigma_{i=1}^{N} G\left(\frac{x-x_i}{\sigma}\right) \cdot x_i}{\Sigma_{i=1}^{N} G\left(\frac{x-x_i}{\sigma}\right)} \] (3.9)

The difference \( m(x) - x \) was later termed as Mean Shift by Fukunaga and Hostetler in their 1975 paper [27]. The authors described this recursive algorithm as a steepest ascent technique that moves the data points closer to their respective modes at each iteration. Later this algorithm became known as the Gaussian Blurring Mean Shift, indicating the successive blurring of the dataset towards its respective nodes. Twenty years later, Cheng in [28] picked up on the concept of Mean Shift and proposed a slight change to the original algorithm. Instead of using the change dataset at each iteration to calculate the mean shift, they chose to use the original dataset, \( X_0 \). The new algorithm became known as Gaussian Mean Shift. [25]
The difference between the GBMS and GMS is that although they both depart from the original, \( X^{(0)} = X_0 \), in GBMS \( X_0 \) is forgotten as the algorithm carries on and the following blurred datasets, \( X^{(1)}, X^{(2)} \) and so on, are created. GMS on the other hand starts by keeping two datasets, \( Y \) and \( X_0 \). Initially \( Y^{(0)} = X_0 \), but at each iteration the changed dataset, \( Y \), is compared to \( X_0 \) to produce the next dataset, \( Y^{(\tau+1)} \) which is translated in Equation 3.10:

\[
m(x) = \frac{\sum_{i=1}^{N} G\left(\frac{\|x_i - x_0\|}{\sigma}\right) \cdot x_{0i}}{\sum_{i=1}^{N} G\left(\frac{\|x_i - x_0\|}{\sigma}\right)}
\]

(3.10)

These algorithms were largely used in applications that dealt with image segmentation and denoising and all sorts of computer vision and image processing applications. Although the interesting applications that benefited from the development of these algorithms, Rao et al posed the following problems: what is it that GBMS and GMS optimize? What changes when going from GBMS to GMS, and vice versa? The authors sought to not only bring a new perspective, from an information theoretic standpoint, to these algorithms, but also to propose a new cost function to develop a broader class of algorithms in which GBMS and GMS are part of. [25, 26]

Starting from the concept of Renyi’s quadratic entropy in subsection 3.2.1 and the PDF estimate given by Equations 3.1 and 3.2, the non-parametric estimator for Renyi’s quadratic entropy is defined as,

\[
H(X) = -\ln V(X)
\]

(3.11)

\[
V(X) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} G\left(\frac{\|x_i - x_j\|}{\sigma'}\right)
\]

(3.12)

Where \( \sigma' = \sqrt{2} \sigma \). The authors expand on this notion and derive the equation for a “cross” entropy between the changed dataset and the original one, \( X_0 \).

\[
H(X, X_0) = -\ln V(X, X_0)
\]

(3.13)

\[
V(X, X_0) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} G\left(\frac{\|x_i - x_{0j}\|}{\sigma'}\right)
\]

(3.14)

The authors also use the Cauchy-Schwarz distance to measure the distance between two PDFs as it has been defined in [29]. Considering \( p(x) \) the PDF of \( X \) and \( q(x) \) the PDF of the original dataset \( X_0 \), this distance is calculated as,

\[
D_{CS}(X, X_0) = \log \left(\frac{\int p^2(x)dx \cdot \int q^2(x)dx}{\left(\int p(x) \cdot q(x)dx\right)^2}\right)
\]

(3.15)

\[
D_{CS}(X, X_0) = -[H(X) + H(X_0) - 2H(X, X_0)]
\]

(3.16)

The authors propose the ITMS with the objective, given a dataset \( X_0 = (x_{0j})_{j=1}^{N} \in R^D \), to find a dataset \( X = (x_i)_{i=1}^{M} \in R^D, M \leq N \) that captures the structure in \( X_0 \). They formulate the idea to
minimize the entropy of the dataset $X$ with constraint that the Cauchy-Schwarz distance is kept to a constant value $k$ with $0 \leq k \leq H(X_0)$. Equation 3.17 shows how this problem can be formulated.

$$\min_{X} H(X) \text{ subject to } D_{cs}(X, X_0) = k \tag{3.17}$$

This constrained optimization can be solved using the Lagrange multiplier as in Equation 3.18.

$$J(X) = \min_{X} H(X) + \lambda \cdot [D_{CS}(X, X_0) - k] \tag{3.18}$$

Expanding this last equation and differentiating $J(X)$ with respect to each $x_i$, the fixed point update rule is given in Equation 3.19.

$$x_{i}^{\tau+1} = \frac{c_1 \sum_{j=1}^{N} G\left( \frac{\|x_i^\tau - x_j^\tau\|^2}{\sigma'} \right) x_j^\tau + c_2 \sum_{j=1}^{N} G\left( \frac{\|x_i^\tau - x_0^\tau\|^2}{\sigma'} \right) x_0^j}{c_1 \sum_{j=1}^{N} G\left( \frac{\|x_i^\tau - x_j^\tau\|^2}{\sigma'} \right) + c_2 \sum_{j=1}^{N} G\left( \frac{\|x_i^\tau - x_0^\tau\|^2}{\sigma'} \right)} \tag{3.19}$$

$$c_1 = \frac{1 - \lambda}{V(X)} \tag{3.20}$$

$$c_2 = \frac{2\lambda}{V(X; X_0)} \tag{3.21}$$

As in GMS, the algorithm is started with $X = X_0$. The authors also note some cases of special interest that arise when choosing different values of the Lagrange Multiplier, $\lambda$.

### 3.2.3.1 When $\lambda = 0$

When choosing this value, the cost function becomes:

$$J(X) = \min_{X} H(X) \tag{3.22}$$

Manipulating this Equation as in the section 3.1 of [25], the fixed point update rule is:

$$x_{i}^{\tau+1} = \frac{\sum_{j=1}^{N} G\left( \frac{\|x_i^\tau - x_j^\tau\|^2}{\sigma'} \right) \cdot x_j^\tau}{\sum_{j=1}^{N} G\left( \frac{\|x_i^\tau - x_j^\tau\|^2}{\sigma'} \right)} \tag{3.23}$$

The authors show that this update rule can also be obtained by changing the value of $\lambda$ in Equation 3.19. Comparing this to GBMS’ expression in 3.9, it is easily concluded that when $\lambda = 0$, ITMS behaves like GBMS. The authors conclude that GBMS tries to minimize the overall entropy of the dataset as it can be seen in Equation 3.22. They also state that the solution is a single point, making the algorithm unstable and when the objective is to find the modes of the PDF, one should be careful in defining the stopping criteria.
3.2.3.2 When $\lambda = 1$

For this value, the cost function is written as:

$$J(X) = \min_X H(X;X_0)$$  \hspace{1cm} (3.24)

Making a similar analysis as in the previous section, the fixed point update rule is:

$$x^{\tau+1}_i = \frac{\sum_{j=1}^N G\left(\frac{\|x^\tau_i - x^\tau_0_j\|}{\sigma} \right) \cdot x^\tau_0_j}{\sum_{j=1}^N G\left(\frac{\|x^\tau_i - x^\tau_0_j\|}{\sigma} \right)}$$  \hspace{1cm} (3.25)

Analysing Equation 3.25, one can easily conclude that in this situation, ITMS behaves like the GMS algorithm. It’s concluded that GMS minimizes the cross entropy $H(X;X_0)$ between the changed dataset $X$ and the original dataset $X_0$. Since the mean shift vector gives direction to the particle with respect to the original PDF, the authors state that this is a stable algorithm to find the modes of a PDF and does not require a special kind of stopping criterion.

3.2.3.3 When $\lambda > 1$

Considering the cost function as in Equation 3.26:

$$J(X) = \min_X H(X) + \lambda \left[D_{CS}(X;X_0) - k\right]$$

$$= \min_X (1-\lambda) H(X) + 2\lambda H(X;X_0) - \lambda H(X_0) - \lambda k$$  \hspace{1cm} (3.26)

For bigger values of $\lambda$, more and more importance is being given to the similarity constraint imposed by the Cauchy-Schwarz distance, $D_{CS}(X;X_0)$. This in turn results in more relevant features of the original dataset being kept. For $1 < \lambda < 2$, the modes give way to the principle curve of the dataset. Increasing the value of $\lambda$ makes the algorithm represent more finely the denser regions in the PDF, and if $\lambda$ is large enough, the resulting dataset will represent all the features of the original dataset.

The authors note that a simple stopping criterion such as the average norm distance of a particle being less than a designated tolerance level, $tol$, for GMS can be implemented. Equation 3.27 shows this stopping criterion.

$$\frac{1}{N} \sum_{i=1}^N \left\| x_i^{(\tau)} - x_i^{(\tau-1)} \right\| < tol$$  \hspace{1cm} (3.27)

However due to the instability of the GBMS algorithm, the authors in [26] propose a different stopping criterion based in Shannon’s Entropy as it can be seen in Equation 3.28.

$$\left| H_S\left(d^{(\tau+1)}\right) - H_S\left(d^{(\tau)}\right) \right| < 10^{-8}$$
Reducing Scenarios - A new approach

\[ H_S(d) = -\sum_{i=1}^{B} f_i \ln f_i \] (3.28)

\( H_S \) represents Shannon’s Entropy, \( f_i \) is the relative frequency of bin \( i \) and the bins span the interval \([0, \max(d)]\) and the number of bins, \( B \), could be chosen as \( B = 0.9N \).

Concluding, the ITMS is a multi-purpose generalized algorithm that has great mode finding capabilities and the potential to be used in several applications. This algorithm is tested in this work to better understand its capabilities and applications to mode finding in a set of forecasting scenarios. The testing procedure and results are featured in Chapter 4.

3.3 Similarity Measures

As defined in [30] similarity is defined as: “Let \( X \) be a set. A function \( s : X \times X \rightarrow \mathbb{R} \) is called a similarity (or proximity) on \( X \) if \( s \) is non-negative, symmetric, and if \( s(x,y) \leq s(x,x) \) holds for all \( x,y \in X \), with equality if and only if \( x = y \)”.

To efficiently cluster the similar scenarios, a good similarity measure must be chosen. In this work, three similarity measures are analysed. The following sections detail the similarity measures used in this work. The scenarios are defined as vectors \( x^{(m)} = \{x^{(m)}_1, \ldots, x^{(m)}_D\} \) in a dataset denoted as \( X = \{x^{(m)}, m = 1, \ldots, M\} \) in which \( D \) is the dimension of the scenarios and \( M \) is the total number of scenarios.

3.3.1 Gaussian Kernel Similarity

Sumaili and Chicco introduced in [24] a similarity measure not based in Euclidean metrics, but based in the concepts of the entropy proposed by Shannon and Renyi in [20, 21].

The similarity between two scenarios, \( i \) and \( j \), in a dataset \( X \), “evaluated in a non-Euclidean space according to the entropy principles”, is expressed in Equation 3.29.

\[ s_{x(i),y(j)} = G \left( x^{(i)} - y^{(j)}, \Sigma_x \right) \]

\[ G \left( x^{(i)} - y^{(j)}, \Sigma_x \right) = \frac{1}{\sqrt{(2\pi)^n det(\Sigma_x)}} \times \exp \left( -\frac{1}{2} (x^{(i)} - y^{(j)})\Sigma_x^{-1} (x^{(i)} - y^{(j)})^T \right) \] (3.29)

\( G \), represents the Gaussian Kernel function described in Equation 3.4 in section 3.2.3 and \( \Sigma_x \) the diagonal covariance matrix by using the points of the scenarios. A simplified Gaussian Kernel like the one used for PDF estimation in section 3.1 could also be used. The authors used this to measure the similarity between two load scenarios.

3.3.2 Cosine Similarity

The cosine similarity, also denominated Orchini Similarity, angular similarity or normalized dot product (internal product), is a similarity on \( \mathbb{R}^n \) defined as the cosine of the angle between two
vectors. The similarity between two vectors is defined in [30] as,

\[ s_{x(i), y(j)} = \cos \phi = \frac{\langle x^{(i)}, x^{(j)} \rangle}{\|x^{(i)}\| \|x^{(j)}\|} \]  

(3.30)

\( \phi \) represents the angle between the two vectors. If the vectors are normalized, this similarity is the internal product between them, hence the name normalized dot product.

### 3.3.3 Kohonen Similarity

Also denominated as the similarity ratio, the Kohonen similarity is a similarity on \( \mathbb{R}^D \) and it’s calculated as in Equation 3.31.

\[ s_{x(i), y(j)} = \frac{\langle x^{(i)}, x^{(j)} \rangle}{\langle x^{(i)}, x^{(j)} \rangle + \|x^{(i)} - x^{(j)}\|^2} \]  

(3.31)

In the special case of the vectors being binary, this similarity is also known as Tanimoto Similarity.

### 3.4 Metaheuristics

Metaheuristics are “general algorithmic frameworks, often nature-inspired, designed to solve complex optimization problems”. In others words, metaheuristics can be applied in a wide array of optimization problems with little modification to solve a specific problem. In recent years, the interest in metaheuristics has risen and are emerging as successful alternatives to more classical approaches. [31, 32]

Metaheuristics have been applied in several decision-making problems in Power Systems as in [33] along with other kinds of applications in different areas present in the book [34].

Considering a finite set \( S \) of feasible solutions \( x \), and a real valued cost function \( F(x) \), a “Deterministic Combinatorial Optimization Problem” can be defined to find:

\[ \min_{x \in S} G(x) \]  

(3.32)

The set \( S \) is denominated as search space. The problem can also feature constraints on the solution. A globally optimal solution is defined as \( x^* \) that \( G(x^*) \leq G(x) \ \forall \ x \in S \). Equation 3.32 is formulated as a minimization problem, but it can also be defined as a maximization problem. The main motivation for the use of these algorithms is that they provide a solution that is as good as possible within reasonable computational time, although not necessarily optimal. To find the exact optimal value, some algorithms may require exponential computation time, making it unpractical for practical purposes. [31]

Some metaheuristics algorithms are: Ant Colonization (AC), Particle Swarm Optimization (PSO), Simulated Annealing (SA), Tabu Search (TS) and Evolutionary Computing (EC) which
Reducing Scenarios - A new approach

divides in: Evolution Strategies (ES), Evolutionary Programming (EP), Genetic Algorithms (GA) and Evolutionary Particle Swarm Optimization (EPSO). [35] A broader scope of the existing meta-heuristics can been at Figure 3.2

![Different classifications of metaheuristics](image)

In this work two metaheuristics were used and compared: Simulated Annealing for a direct approach and Evolutionary Programming for a population-based approach. The two algorithms are detailed in the following subsections.

### 3.4.1 Simulated Annealing

Simulated Annealing is based on a model developed by Metropolis et al in [36]. This model intends to simulate the annealing process of metal in which the particles arrange themselves in a thermal equilibrium, i.e. a state of minimum energy. [31, 35]

The algorithm is based on the principle of local search heuristics in a predefined neighbourhood structure on the search space $S$. There is also a control parameter that is denominated as temperature like in the model that it is based in. At each iteration, a neighbour solution $y$ to the present solution $x$ is found. At this point both solutions are compared according to their objective
function, if \( y \) has the better objective function value, then \( y \) is accepted as the new optimal value. If \( y \) has not a better value, then \( y \) is only accepted as the new optimal value with a certain probability depending on the temperature and the difference between the objective function of the two solutions.

A skeleton for this algorithm can be seen below taken from [31] and is shown in Algorithm 1.

**Algorithm 1** Simulated Annealing (SA)

1: Initialize state \( x \) and temperature parameter \( T_1 \)
2: for iteration \( k = 1, 2, \ldots \) to Max number of iterations do
3: select \( y \) randomly from \( S(x) \)
4: if \( G(y) \leq G(x) \) then
5: set \( x = y \)
6: else if \( \exp \left( \frac{G(x) - G(y)}{T_k} \right) \leq \text{uniform}[0, 1] \) then
7: set \( x = y \)
8: end if
9: end for

From this skeleton it should be noted that, \( S \) represents the search space, \( G(x) \) the objective function from Equation 3.32, \( T \) represents the temperature value that decreases according to a predefined cooling schedule, \( S(x) \) represents the neighbourhood structure and is contained in space \( S \) to create the neighbour solutions, lastly \( \text{uniform}[\alpha, \beta] \) represents a procedure to return a uniformly distributed pseudo-random number from the interval \( [\alpha, \beta] \), usually \( [0, 1] \).

The SA metaheuristic was firstly introduced in the area of combinatorial optimization by Kirkpatrick et al. in their paper of 1983 [37]. In [38], it can be seen the application of SA to an expansion planning of a transmission network and the calculation of long-term marginal prices. The SA algorithm explained in [38] served as the template for the SA algorithm used in this work.

### 3.4.2 Evolutionary Programming

Evolutionary Computation, in which EP is inserted, takes cues from biological evolution paradigms but it’s not restricted by some biological realizations. These algorithms, as the name suggests, are based in Darwin’s theory of evolution, in which mutations command the evolution of species. An evolutionary algorithm finds an optimal solution by creating a “population” that represents a set of solutions to the problem. Each “individual”, a solution in the population, is evaluated and the best are selected for “reproduction”. A “new generation” of individuals is produced from the original population which are in turn evaluated to find the best individuals for reproduction. This process repeats generation after generation. The population is, iteratively, comprised of better individuals until a stopping criterion is met. The best individual is then selected as the optimal solution. [39]

EC algorithms distinguish themselves in terms of the representation of the solution, selection procedure or the reproduction procedure. EP and ES fall in the sub-category of “phenotypic” algorithms as the representation of the solutions is based in the variables of the problem, not requiring any type of coding and decoding algorithm. On the other hand, GA is a “genotypic” algorithm as it requires the representation of the solutions in sequences called “chromosomes”,...
just as the genetic representation in living beings. These chromosomes are created using coding algorithms that represent the characteristics of the variables of the problem. [39]

In this work, of the EC metaheuristics, EP was the chosen method and more focus will be given to it.

EP was originally proposed by Fogel et al in 1966 in [40]. EP is an algorithm in which an initial population of \( \lambda \) individuals is randomly created and evaluated. Then this population is cloned and later mutated. The mutation scheme is usually done by adding to the cloned individual a random disturbance. Be it an individual \( X^{(g)} \) a vector with \( n \) characteristics, or variables, so that \( X^{(g)} = X_1^{(g)}, \ldots, X_n^{(g)} \) in a generation \( g \) and the random disturbance \( Z \), the new individual \( \tilde{X} \) is defined as,

\[
\tilde{X} = X^{(g)} + Z
\]

\[
Z = \sigma \times (N_1(0,1), \ldots, N_n(0,1))
\]

\( N_j(0,1) \) represents the value obtained by a Gaussian Distribution with a mean valued at zero and a variance valued at one in variable \( j \), \( \sigma \) represents the value of the mutation rate. This mutation rate should not be a fixed value if one desires a fast convergence. One can adopt the Self-Adaptive strategy to control the mutation rate. In this strategy the mutation rate also suffers a mutation, this one being multiplicative as in,

\[
\sigma^{(g+1)} = \xi \times \sigma^{(g)}
\]

The random variable \( \xi \) should be obtained from a Gaussian distribution with mean valued at zero and variance at 1,

\[
\xi = 1 + \tau N(0,1)
\]

Equation 3.36 introduces a factor to control the evolution of the mutation rate, \( \tau \) and it’s usually called learning factor. This factor is essential as it conditions the velocity and precision of this adaptive strategy.

The selection method usually used in EP is a stochastic tourney. The simplest tourney is the one in which there is a random choice of pairs of individuals. They are compared in regard to their value of the objective function and the selection of the best individual is done with a certain probability, usually great but inferior to 1. This process is repeated until the number of individuals in the population is met. This population will be the progenitor of the next generation. [39]

The process of cloning, mutating the population and selecting the individuals is done in every iteration until a stopping criterion is met. Usually a pre-determined number of generations is used as the stopping criterion.

A skeleton for an EP algorithm is described below taken from [35] in Algorithm 2.

The next section details the proposed methodology along with the formulation of metaheuristics used.
3.5 A New Clustering Technique

The proposed methodology sets out to solve the problem of not representing strange low represented scenarios that could have significant impact in case they occur, in a scenario reduction procedure.

To achieve an efficient scenario reduction while still maintaining the representation of the strange scenarios, a clustering technique based in an optimization problem is proposed: find, around a fixed number of representative scenarios, the scenarios that create these clusters while maximizing the Entropy between the clusters and minimizing the sum of the entropies in each cluster, concepts that were introduced in subsections 3.2.1 and 3.2.2. The Entropy between clusters measures the divergence between the clusters and the entropy in a cluster measures the divergence between the elements of the cluster. Maximizing the entropy between clusters will result in a set of clusters that are the most different from each other and minimizing the entropy in each cluster will create clusters that group the most similar ones and leaving the ones that can’t be clustered in its own cluster, the Entropy in a cluster is lowest when it has only one scenario. Combining this two criteria will present a set of clusters that differ from each other as most as possible, but each cluster will have the most similar scenarios in them, with a few clusters with the more strange scenarios.

This requires clustering the scenarios that are similar to another in accordance to a chosen similarity metric. Having recognized the problem that choosing the similarity metric poses in the reviewed literature, the metrics detailed in section 3.3 will be compared.

The optimization problem can be solved by any optimization algorithm and the two metaheuristics detailed in section 3.4 were adapted to suit the needs of this methodology.

3.5.1 Scenario and Cluster definitions

The data to be analysed and clustered are forecast scenarios. This work focuses both on wind power and load forecast scenarios. Dr. Sumaili provided the sets of scenarios. Wind power scenarios were created under a Monte Carlo sampling process as in [15]. Considering that this dataset contains $M$ scenarios such that,

$$x^{(m)} = \{x_1^{(m)}, \ldots, x_T^{(m)} \}$$

(3.37)
Reducing Scenarios - A new approach

For a time horizon \( t = 1, \ldots, T \) in scenario \( m = 1, \ldots, M \), with \( T \) being the dimension of the vectors containing the scenario data. For this work, the vectors are represented hourly for a given day, meaning that \( T = 24 \) and they are normalized in respect to the installed power of the wind farm, meaning that the components’ range is \([0, 1]\). [15]

On the other hand, the load scenarios are comprised of 96 points, \( T = 96 \), meaning that the forecast is done for 15-min data. The vectors are normalized to reference power value, this time being the maximum power of the daily load scenario. [24]

The clustering procedure groups the \( M \) scenarios in \( K \) clusters so that \( C^{(1)} \subset X \), for \( k = 1, \ldots, K \) and \( 1 \leq K \leq M \). Considering \( N^{(k)} \) the number of scenarios in \( C^{(k)} \), the following must be verified, \( N^{(1)} + \ldots + N^{(K)} = M \).

### 3.5.2 Objective Function

As it was previously stated, this clustering technique is based in an optimization problem. The objective function for this problem is illustrated in Equation 3.38, considering the \( K \) clusters of scenarios \( C^{(1)} \ldots C^{(K)} \).

\[
\max O(C^{(1)} \ldots C^{(K)}) = H(C^{(1)} \ldots C^{(K)}) - \sum_{i=1}^{K} h(C^{(i)})
\] (3.38)

\( H(C^{(1)} \ldots C^{(K)}) \) represents the entropy between clusters as shown in equation 3.8 \( h(C^{(i)}) \) represents the entropy of a cluster \( i \) calculated as in Renyi’s quadratic Entropy in Equation 3.6. As this is a multi-criterion optimization problem, a weight \( \lambda \) as introduced first as in Equation 3.39 and later as in Equation 3.40 so that when \( \lambda = 0 \), the objective function only maximizes the entropy between clusters and when \( \lambda = 1 \), the sum of the entropies of each cluster is minimized.

\[
\max O(C^{(1)} \ldots C^{(K)}, \lambda) = H(C^{(1)} \ldots C^{(K)}) - \lambda \times \sum_{i=1}^{K} h(C^{(i)})
\] (3.39)

\[
\max O(C^{(1)} \ldots C^{(K)}, \lambda) = (1 - \lambda) \times H(C^{(1)} \ldots C^{(K)}) - \lambda \times \sum_{i=1}^{K} h(C^{(i)})
\] (3.40)

### 3.5.3 Optimization Algorithms

Both metaheuristics used in this work needed to be adapted for this optimization problem. As it stands, the optimization problem is a combinatorial problem, i.e. for a desired \( k \) number of representative scenarios, find the \( k \) that best aggregate the scenarios into clusters. In this section the adaptations for both metaheuristics are detailed.

### 3.5.3.1 Evolutionary Programming

The classical EP formulation to introduce small mutations to the variables of an individual may lead to highly irregular curves in the context of this problem. So a new way to apply the mutations was introduced to find a scenario to work as an aggregator.
3.5 A New Clustering Technique

The EP algorithm is started by defining the following variables: maximum number of generations $N_{\text{gen}}$, number of individuals in a population $N_{\text{ind}}$, number of representative scenarios $N_{\text{scenarios}}$, the reference probability $p_{\text{ref}}$ for the stochastic tourney and the weight $\lambda$. Then the user chooses the scenario dataset to reduce, load or wind power and the type of similarity to use. Upon choosing the type of similarity, a similarity matrix is calculated to compare each and every scenario in the dataset. To aid the choice of new scenarios in the mutation process, a probability distribution (PD) is created to weigh the probability of a scenario being chosen based in the similarity results. Considering a $M \times M$ similarity matrix $S$, the PD is calculated as in Equation 3.41 for a scenario $i$.

$$p(S(i, j)) = \frac{S(i, j)}{\sum_{j=1}^{M} S(i, j)} \quad \text{for } j = 1, ..., M \quad (3.41)$$

A vector containing all of the $p(S(i, j))$ is then sorted in a descending order. A cumulative distribution function (CDF) is then created as in Equation 3.42.

$$c_k(S(i, j)) = \sum_{j=1}^{k} p(S(i, j)) \quad \text{for } k = 1, ..., M \quad (3.42)$$

The result is a CDF like the one in Fig. 3.3 for each scenario. The use of this cumulative function favours the change to more similar scenarios as they a have a higher probability to be chosen.

![Cumulative Distribution Function](image)

Figure 3.3: A cumulative distribution function

The next step is to generate an initial population: for each individual according the desired number of representative scenarios $K$, $K$ scenarios are chosen randomly to create the clusters. The creation of the clusters is done according to the similarity matrix calculated beforehand and the scenarios that are more similar to the aggregator scenario are grouped in each cluster. After all
of the clusters are created, the individual is evaluated by calculating the objective function. An individual’s characteristics are: the aggregator scenarios, their indexes according to the scenario dataset and the respective clusters. The entropy between clusters, a vector with entropy for each cluster, the sum of these entropies and objective function are the characteristics of the evaluation function for the solution. Below, Algorithm 3 explains how to create the first population.

Algorithm 3 Generate Initial Population
1: Read the problem’s variables
2: for iteration \( i = 1, 2, \ldots \) to Number of Individuals do
3: randomly select \( k \) scenarios
4: create the clusters for each aggregator scenario
5: calculate the entropy between clusters
6: calculate the entropy for each cluster
7: calculate the objective function
8: save the individual’s characteristics and evaluation results in structure
9: end for

After the creation of the initial population, an iterative process begins. For each generation a second population with equal number of individuals is created. This time the aggregator scenarios are not randomly chosen, they are in fact chosen with a given probability. For each individual and for each cluster, this probability is randomly calculated and compared the previous scenario’s CDF to choose the next scenario. After choosing all of the new scenarios, the clustering and evaluation process begins. Algorithm 4 details this process.

Algorithm 4 Generate Next Population
1: Read the problem’s variables and the previous population’s individuals.
2: for iteration \( i = 1, 2, \ldots \) to Number of Individuals do
3: Read the aggregator scenarios indexes
4: for iteration \( j = 1, 2, \ldots \) to Number of representative Scenarios do
5: Randomly assign a probability
6: Find the closest CDF value for the aggregator scenario of the previous individual
7: Return the corresponding index
8: if the scenario was previously chosen then
9: Run the selection process again, return to 5
10: else
11: Continue
12: end if
13: end for
14: Build the Clusters
15: Calculate the entropy between clusters and the entropy for each cluster
16: Calculate the objective function
17: Save the individual’s characteristics and evaluation results in structure
18: end for

Then the two populations, the previous and the new one are joined into a single one to find the individuals that will be present in the next generation. This selection process is done with a
modified stochastic tourney. Unlike the one described in subsection 3.4.2, in this one the individual with the highest value for the objective function is always selected with the remaining individuals being selected by the classical stochastic tourney. The reason for this choice was that in previous uses of the classical stochastic tourney, the best individual up to a certain point could be lost and resulting in non-monotonic behaviour of the objective function along the algorithm’s run. One could opt for an elitist selection method in which only the best individuals are carried to the next generation. This could make the algorithm stop in local good points but not the maximum value. The algorithm for the stochastic tourney is presented in Algorithm 5.

Algorithm 5 Stochastic Tourney
1: Sort the joined populations in descending order
2: Save the first one, with the highest value, in a new population
3: for iteration $i = 2, \ldots$ to Number of Individuals do
4: Randomly choose to two individuals
5: Choose a random value, $p$, between 0 and 1
6: Identify the better individual
7: if $p < P_{ref}$ then
8: The better individual is chosen
9: else
10: The worse individual is chosen
11: end if
12: Remove both individuals from the set
13: end for
14: The new population is used in the next generation

The probability reference, $P_{ref}$, should have a high value, but inferior to 1. In this work 0.85 was used. The process described up to here is repeated until the stopping criterion is met. In this work, a maximum number of generations was used. The full algorithm for the EP procedure is featured in Algorithm 6.

Algorithm 6 Full EP Algorithm
1: Define $N_{gen}, N_{ind}, N_{scenarios}, \lambda, P_{ref}$
2: Choose the scenario dataset, load or wind forecast scenarios
3: Choose the similarity measure: Gaussian Kernel, Cosine or Kohonen
4: Calculate the similarity matrix of the scenarios dataset
5: Build the PD and CDF for each scenario
6: Generate the first population using Algorithm 3
7: Determine the best individual of the initial population
8: Create a vector, $V$, to save the best objective function values for each generation
9: Save the objective function value of the best individual in $V$
10: for count$_{gen} i = 1, \ldots$ to $N_{gen}$ do
11: Apply the "mutation" procedure in Algorithm 4
12: Apply Stochastic Tourney in Algorithm 5
13: Save the objective function value of the best individual in $V$
14: end for
15: Retrieve the results
3.5.3.2 Simulated Annealing

As it was said before, the structure of the SA algorithm closely follows the one in [38]. The main variables to be chosen by the user in this algorithm are the number of representative scenarios, the weight $\lambda$, and variables to control the stopping criterion, a starting temperature $T$, a temperature index $T_{\text{index}}$, a maximum number of worst solutions $WSC_{\text{max}}$, a maximum number of same temperature iterations $STC_{\text{max}}$ and maximum number of iterations $IC_{\text{max}}$.

Similarly to the EP algorithm, this one starts by choosing the scenario dataset and the similarity type. The similarity matrix is calculated and so is the CDF matrix. An iteration counter (IC), worst solution counter (WSC), same temperature counter (STC) are created. This application will be explained later.

Then the first solution is created. The process is equal to the one of generating an initial population in EP with the main difference that only one solution is created. Algorithm 7 describes this process:

**Algorithm 7 Initial Solution**

1: Read the problem’s variables
2: randomly select $k$ scenarios
3: create the clusters for each aggregator scenario
4: calculate the entropy between clusters
5: calculate the entropy for each cluster
6: calculate the objective function
7: save the individual’s characteristics and evaluation results in structure

After the creation of the initial solution, an iterative process begins. In each iteration a new neighbour solution is created. Again the creation of this neighbour solution is similar to the “mutation” process in EP. The procedure for this case is shown in Algorithm 8.

**Algorithm 8 Generate Neighbour Solution**

1: Read the problem’s variables and the previous population’s individuals.
2: Read the aggregator scenarios indexes
3: for iteration $j = 1, 2, \ldots$ to Number of Representative Scenarios do
4: Randomly assign a probability
5: Find the closest CDF value for the aggregator scenario of the previous individual
6: Return the corresponding index
7: if the scenario was previously chosen then
8: Run the selection process again, return to 4
9: else
10: Continue
11: end if
12: end for
13: Build the Clusters
14: Calculate the entropy between clusters and the entropy for each cluster.
15: Calculate the objective function
16: Save the solutions’s characteristics and evaluation results in structure
The new solution and the best one are then compared. If the new solution has a higher objective function value, then the new solution is chosen as the new best value and the current value. If the new solution isn’t better, then this solution goes through a selection process. The new solution is chosen as the current value if a randomly generated probability is inferior to a given probability reference that is calculated for each iteration as in Equation 3.43.

\[ p_{ref} = e^{\frac{C(X_{NEW}) - C(X_{CURRENT})}{K_B \times T}} \]  

(3.43)

\( C(X_{NEW}) \) is the objective function’s value of the new solution, \( C(X_{CURRENT}) \) is the objective function’s value of the current solution, \( K_B \) is a Boltzmann’s Constant for physical processes and \( T \) is the current temperature. The Boltzmann’s Constant in this context, as stated in [35] hasn’t a referenced value and it must be calculated so that in the first selection \( p_{ref} = 0.5 \). Equation 3.44 shows this.

\[ K_B = \frac{C(X_{NEW}) - C(X_{CURRENT})}{\ln(p_{ref}) \times T} \]  

(3.44)

If the randomly drawn probability is inferior to the probability reference, then the new solution is chosen as the current solution and the WSC is incremented. The next step is to update the temperature according to the cooling schedule. First, if the STC is inferior to the \( STC_{max} \), the counter STC is incremented. If not, the temperature \( T \) is reduced by \( T = T \times T_{index} \) and STC reverts back to 1.

To stop the SA algorithm there are three criteria: a maximum number of iterations or a maximum number of worst solutions or the temperature reaches the minimum temperature. The algorithm stops if the \( IC_{max} \), or the \( WSC_{max} \) or if \( T \) falls below of the minimum temperature, which in this work is 1. Algorithm 9 shows the full SA algorithm.

Both algorithms were implemented in Matlab, along with side functions to save the testing data in Excel spreadsheets and plot the resulting clusters.

### 3.5.4 Choosing the Representative Scenario

When the either the EP or SA algorithms have found an optimal solution, by identifying a cluster and its members that maximize the objective function, a representative scenario for each cluster must be found. Papers [15, 16] propose three ways to identify a representative scenario: choosing the aggregator scenario as the representative scenario; calculate the mean value of the cluster by averaging its members to get the centroid of the cluster; use the ITMS algorithm to find the mode within the clusters.

They note that the latter cases produce an artificial scenario, while the first option may cause the total expected value to "deviate from the expected value from the whole set of scenarios". In this work, the first option was used.
Algorithm 9 Full SA Algorithm

1: Define $N_{\text{scenarios}}, \lambda, T, T_{\text{index}}, WSC_{\text{max}}, STC_{\text{max}}$ and $IC_{\text{max}}$.
2: Choose the scenario dataset, load or wind forecast scenarios
3: Choose the similarity measure: Gaussian Kernel, Cosine or Kohonen
4: Calculate the similarity matrix of the scenarios dataset
5: Build the PD and CDF for each scenario
6: $WSC = 0, STC = 1, IC = 1, T_{\text{min}} = 1$
7: Generate the first solution using Algorithm 7 and save it as $C_{\text{BEST}}$ and $C_{\text{CURRENT}}$
8: Create a vector, $V$, to save the best objective function values for each iteration
9: $\text{endcheck} = 0$
10: while $IC < IC_{\text{max}}$ and $\text{endcheck} \neq 1$ do
11: Create new solution using Algorithm 8
12: Compare the two solutions
13: if $C_{\text{NEW}} > C_{\text{CURRENT}}$ then
14: Save the new solution as $C_{\text{BEST}}$ and $C_{\text{CURRENT}}$ and reset $WSC$
15: else
16: if first selection then
17: $p_{\text{ref}} = 0.5$ and calculate $K_B$ using Equation 3.44
18: else
19: Calculate $p_{\text{ref}}$ using Equation 3.43
20: end if
21: Randomly assign a probability $p$
22: if $p \leq p_{\text{ref}}$ then
23: Set $C_{\text{NEW}}$ as $C_{\text{CURRENT}}$
24: end if
25: Increment $WSC$
26: end if
27: if $WSC > WSC_{\text{max}}$ then
28: $\text{endcheck} = 1$
29: end if
30: if $STC > STC_{\text{max}}$ then
31: Reduce according to the cooling schedule, $T = T \times T_{\text{index}}$
32: Set $STC = 1$
33: else
34: Increment STC
35: end if
36: if $T < T_{\text{min}}$ then
37: $\text{endcheck} = 1$
38: end if
39: Increment $IC$
40: Save the objective function value of the best solution in $V$
41: end while
Chapter 4

Results in Study Cases

This chapter features the results that were obtained in the tests using the methodologies presented in the previous chapter. During the learning process for this Dissertation, several concepts were studied and tested. With that in mind, this chapter presents in the first section all of the datasets used in this Dissertation, the second section details the testing of the PDF estimator using the Pazen Windows Technique and the third section features the test results of the ITMS algorithm for wind and load scenarios. Section 4.4 features the results of the proposed methodology for wind power forecast and load forecast.

4.1 Used Datasets

In the course of this work, several datasets were used. Some were created to test the application of the PDF estimator and the ITMS algorithm. Wind power and load forecast scenarios were given by Dr. Jean Sumaili. They were initially used to find the modes using the ITMS algorithm and later for the forecasting tests. The datasets are then shown in the next Figures.

![Double Crescent shape curves - Dataset 1](image)

Figure 4.1: Double Crescent shape curves - Dataset 1
Results in Study Cases

Figure 4.2: Random Gaussian Clusters - Dataset 2

Figure 4.3: Random Gaussian Clusters with wider dispersion - Dataset 3

These datasets were used in PDF estimation and testing the ITMS algorithm. The main reason to create these datasets was to obtain similar results to the ones obtained by Rao et al in [25] with Dataset 1 and in [26] with Dataset 2 and Dataset 3.

The load and wind power forecast scenarios were obtained using the procedure described in Section 3.5.1. The first wind power scenarios that were given for testing contain the forecasting scenarios for 3 days with 300 scenarios for each day. Each scenario contains 24 points and each point contains the mean wind power production for each hour in p.u. (per unit). From these scenarios, the scenarios of day 2 in Figure 4.5 were chosen to test the scenario reduction algorithms. The load scenario dataset is comprised of 193 scenarios and is shown in Figure 4.6 and were used in its entirety.
4.1 Used Datasets

Figure 4.4: Wind Power forecast scenarios for 3 days

Figure 4.5: Wind Power forecast scenarios for 1 day

Figure 4.6: Load forecast scenarios for 1 day
4.2 Study Case 1: PDF estimation using Parzen Windows technique

To better understand the concept of PDF estimation using the Parzen Window technique, a function in Matlab to estimate the PDF in a given point in space was created. This function is called “parzen_estimate” and has the following input and output arguments:

- **Input arguments:**
  - $x_i$ – dataset to be used in the estimation;
  - $d$ – the point in space to estimate the PDF in;
  - $bw$ – bandwidth of the Gaussian kernel, $\sigma$.

- **Output arguments:**
  - $pdf$ – the PDF estimation in the point $d$.

To apply this function to all the space of the dataset, the following script “estimate” was created:

```
Algorithm 10 Estimate PDF
1: clear x,y,pdf
2: x = −7 : 0.1 : 4; y = −7 : 0.1 : 4, c = 0
3: for iteration i = 1 : length(x) do
4:   for iteration j = 1 : length(x) do
5:     pdf(i, j) = parzen_estimate(dataset1, [x(j), y(j)], sqrt(\sigma^2));
6:   c = c + 1;
7: end for
8: end for surf(x,y,pdf)
```

The script in Algorithm 10 was applied to Dataset 1 with the conditions stated below:

$$\sigma^2 = 0.05$$
$$x = −7 : 0.1 : 4$$
$$y = −7 : 0.1 : 4$$

For Dataset 2 and Dataset 3, the bandwidth and the following space were used:

$$\sigma^2 = 0.004$$
$$x = −0.2 : 0.01 : 1.2;$$
$$y = −0.2 : 0.01 : 1.2;$$

The surface and contour plots of the resulting PDFs are shown in Figures 4.7 through 4.12.
4.2 Study Case 1: PDF estimation using Parzen Windows technique

Figure 4.7: Contour plot of the PDF for dataset 1

Figure 4.8: Surface plot of the PDF for dataset 1
Results in Study Cases

Figure 4.9: Contour plot of the PDF for dataset2

Figure 4.10: Surface plot of the PDF for dataset2
4.2 Study Case 1: PDF estimation using Parzen Windows technique

Figure 4.11: Contour plot of the PDF for dataset3

Figure 4.12: Surface plot of the PDF for dataset3
The developed procedure successfully estimated the PDF of each dataset. Having just the datasets and no more information about it or its PDF, by choosing an appropriate bandwidth for the kernel functions with the Parzen Window technique, one can estimate an accurate representation of the probability density function. These results were later used to compare ITMS’s mode finding results to validate the results of the developed programs.

### 4.3 Study Case 2: ITMS algorithm

The purpose of this study case is to evaluate ITMS mode finding capabilities and the influence that the parameters $\lambda$ and $\sigma$ have in the workings of this algorithm. The MATLAB function created to apply the ITMS algorithm has the following input and output arguments:

- **Input arguments:**
  - $X_0$ – original dataset;
  - $l$ – Lagrange Multiplier;
  - $bw^2$ – square value of the bandwidth of the Gaussian kernel $\sqrt{\sigma}$;
  - $tol$ – tolerance of the stopping criterion.

- **Output arguments:**
  - $X_f$ – the variable that returns the result of the algorithm;
  - count – iteration counter.

It was also created a labelling process for use in mode finding. This labelling process identifies the clusters created and the elements of the original dataset that belong to them. They were applied while testing the ITMS algorithm to Datasets 2 and 3 and the forecast scenarios.

#### 4.3.1 Testing with Datasets 1, 2 and 3

The tests conducted had the objective to study and confirm the influence of the choice of the values of $\lambda$ and the kernel bandwidth - $\sigma$. Table 4.1 shows the conditions of each test conducted and the number of iteration it took to converge. For Dataset 1 a tolerance level of $10^{-3}$ was used and for the other two, a tolerance level of $10^{-6}$.

Given the high amount of data and plots generated in these tests, they will not be shown in this section but the reader may find them in Appendix A. Test number 2 with dataset 3 did not converge. In figures 4.13 through 4.15 one can see the results of these tests.
## 4.3 Study Case 2: ITMS algorithm

### Table 4.1: Testing conditions and number of iterations for each Dataset

<table>
<thead>
<tr>
<th>No.</th>
<th>D.1</th>
<th>Iter.</th>
<th>D.2 and D.3</th>
<th>Iter. for D.2</th>
<th>Iter. for D.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\lambda = 0$, $\sigma^2 = 0.50$</td>
<td>12</td>
<td>$\lambda = 0$, $\sigma^2 = 0.010$</td>
<td>177</td>
<td>126</td>
</tr>
<tr>
<td>2</td>
<td>$\lambda = 0$, $\sigma^2 = 0.10$</td>
<td>12</td>
<td>$\lambda = 0$, $\sigma^2 = 0.005$</td>
<td>84</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>$\lambda = 0$, $\sigma^2 = 0.01$</td>
<td>33</td>
<td>$\lambda = 0$, $\sigma^2 = 0.001$</td>
<td>31</td>
<td>78</td>
</tr>
<tr>
<td>4</td>
<td>$\lambda = 1$, $\sigma^2 = 0.50$</td>
<td>202</td>
<td>$\lambda = 1$, $\sigma^2 = 0.010$</td>
<td>101</td>
<td>125</td>
</tr>
<tr>
<td>5</td>
<td>$\lambda = 1$, $\sigma^2 = 0.10$</td>
<td>133</td>
<td>$\lambda = 1$, $\sigma^2 = 0.005$</td>
<td>58</td>
<td>62</td>
</tr>
<tr>
<td>6</td>
<td>$\lambda = 1$, $\sigma^2 = 0.01$</td>
<td>32</td>
<td>$\lambda = 1$, $\sigma^2 = 0.001$</td>
<td>90</td>
<td>53</td>
</tr>
<tr>
<td>7</td>
<td>$\lambda = 2$, $\sigma^2 = 0.50$</td>
<td>62</td>
<td>$\lambda = 2$, $\sigma^2 = 0.010$</td>
<td>817</td>
<td>495</td>
</tr>
<tr>
<td>8</td>
<td>$\lambda = 2$, $\sigma^2 = 0.10$</td>
<td>67</td>
<td>$\lambda = 2$, $\sigma^2 = 0.005$</td>
<td>62</td>
<td>165</td>
</tr>
<tr>
<td>9</td>
<td>$\lambda = 2$, $\sigma^2 = 0.01$</td>
<td>34</td>
<td>$\lambda = 2$, $\sigma^2 = 0.001$</td>
<td>402</td>
<td>82</td>
</tr>
<tr>
<td>10</td>
<td>$\lambda = 10$, $\sigma^2 = 0.50$</td>
<td>65</td>
<td>$\lambda = 10$, $\sigma^2 = 0.010$</td>
<td>206</td>
<td>296</td>
</tr>
<tr>
<td>11</td>
<td>$\lambda = 10$, $\sigma^2 = 0.10$</td>
<td>37</td>
<td>$\lambda = 10$, $\sigma^2 = 0.005$</td>
<td>211</td>
<td>358</td>
</tr>
<tr>
<td>12</td>
<td>$\lambda = 10$, $\sigma^2 = 0.01$</td>
<td>28</td>
<td>$\lambda = 10$, $\sigma^2 = 0.001$</td>
<td>594</td>
<td>244</td>
</tr>
</tbody>
</table>

Table 4.2: Final results of the mode finding to each dataset.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Conditions</th>
<th>Number of iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>D.1</td>
<td>$\lambda = 1$, $\sigma^2 = 0.050$</td>
<td>92</td>
</tr>
<tr>
<td>D.2</td>
<td>$\lambda = 1$, $\sigma^2 = 0.004$</td>
<td>22</td>
</tr>
<tr>
<td>D.3</td>
<td>$\lambda = 1$, $\sigma^2 = 0.004$</td>
<td>57</td>
</tr>
</tbody>
</table>

Figure 4.13: Mode finding results (right) compared to the PDF of Dataset 1 (left)
Regarding the first batch of tests conducted, several observations may be made about the values of the Lagrange multiplier $\lambda$ and the kernel’s bandwidth $\sigma$.

As it was said before, for $\lambda = 0$ the Algorithm behaves like the GBMS algorithm. In all datasets and for any value of bandwidth the GBMS exhibited poor mode finding capabilities as in Figure 4.16. As explained in [25, 26] GBMS is an unstable algorithm and if the interest is to find the modes, one should take care in stopping the algorithm at the appropriate iteration. Generally, this algorithm converged in less iterations that any other.
When $\lambda = 1$, the algorithm behaves as the GMS algorithm. In this situation choosing the correct value of the kernel bandwidth will determine this algorithm’s accurate mode finding. Given the right value of $\sigma$, this algorithm can find the modes correctly and can be used for clustering procedures.

Regarding Dataset 1, if one’s purpose is to find the principal curve of the data, setting the $\lambda$ value between 1 and 2 will achieve the desired results as one can see in Figure 4.17.

Further increasing $\lambda$ will result in the principal features of the dataset being kept and in the extreme scenario, for a high enough value of $\lambda$, the result will match the original dataset.
4.3.2 Mode finding in wind power forecast scenarios

As proved in previous tests, the ITMS algorithm exhibits great mode finding capabilities. In these tests the ITL Mean Shift Algorithm was used to find the modes of Dataset 1 which is comprised
4.3 Study Case 2: ITMS algorithm

of 900 Wind Power Forecast Scenarios from 3 different days. All the scenarios were scrambled and it was impossible to know to which day the scenario belonged.

The objectives of these tests were:

- To find the modes of the dataset and to show that the actual distribution represented by the scenario set is unimodal;

- To cluster the scenarios in its respective day – to create 3 clusters that contain the scenarios that belong in a given day.

As in [15] the ITMS was run with $\lambda = 1$ and $\sigma = 0.18$, with tolerance $10^{-6}$ the algorithm converged in 34 iterations. It can be clearly seen in Figure 4.20, that the ITMS algorithm found 3 modes, one for each day. This result shows that for each day, the structure of the scenarios is unimodal.

![Figure 4.20: The modes of the dataset with wind power forecast scenarios](image)

Using the developed classification procedure to cluster the scenarios to its respective mode, one gets the scenarios for each day. Figures 4.21 through 4.23 illustrate this. The use of this algorithm and the classification procedure made possible finding the scenarios that belong in each day in both cases.
Figure 4.21: Day 1 wind power forecast scenarios and its mode – 301 scenarios

Figure 4.22: Day 2 wind power forecast scenarios and its mode – 300 scenarios
4.4 Study Case 3: Scenario Reduction

Although the initial dataset was composed of 300 scenarios for each day, there was one scenario that was misclassified. This error is acceptable as this misclassification arises from the use of a single kernel size. Rao et al stated in [26] in their tests with GMS, the way that ITMS behaves when $\lambda = 1$, there were a few misclassified points in their testing of the Random Gaussian Kernels that arose from the points being located in the narrow valleys between two clusters of their PDF representation and their gradient directions give way to the incorrect mode. The same principle applies to this situation.

4.4 Study Case 3: Scenario Reduction

In this study case, the application of the new scenario reduction methodology is studied. Figure 4.6 represents the load scenarios dataset that was used. To test the scenario reduction procedures in a set of wind power forecast scenarios, Day 2 that resulted of the daily classification done with ITMS in Section 4.3.2 was chosen as it was the day with apparently all of the scenarios correctly identified. Figure 4.5 shows the set of scenarios to be reduced.

Preliminary tests showed that both criteria had values that are very far apart, for example in one test, $H(C^{(1)} \ldots C^{(K)}) = -20.03$ and $\sum_{i=1}^{K} h(C^{(i)}) = -1419.96$. Looking at the objective function in Equation 3.40, it’s easily concluded that the second criterion "weighs" more than the first, resulting in a higher minimization of the value of $\sum_{i=1}^{K} h(C^{(i)})$ than the maximization of $H(C^{(1)} \ldots C^{(K)})$. If a criterion has an higher "weight" than the other, then the objective function will evolve in a different manner from what it’s wanted. To cope with situation, the second criterion is scaled down by a factor of 100 and $\lambda$ is tuned so that the objective function evolves by increasing the value of $H(C^{(1)} \ldots C^{(K)})$ and decreasing the value of $\sum_{i=1}^{K} h(C^{(i)})$.
4.4.1 Performance Tests

To evaluate the performance of the metaheuristics 2 tests were carried out, one for each scenario set for each metaheuristic. The parameters to evaluate were: value of the objective function, the behaviour of the objective function during the tests and the total run time. The conditions for the conducted tests for SA and EP are presented in Tables 4.3 and 4.7 for 25 representative scenarios.

Table 4.3: The conditions for the tests conducted using SA

<table>
<thead>
<tr>
<th>Set</th>
<th>$T_{ini}$</th>
<th>$T_{min}$</th>
<th>$T_{index}$</th>
<th>$WSC^{max}$</th>
<th>$STC^{max}$</th>
<th>$IC^{max}$</th>
<th>Similarity</th>
<th>$\lambda$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wind</td>
<td>1000</td>
<td>1</td>
<td>0.99</td>
<td>75000</td>
<td>1000</td>
<td>375000</td>
<td>Gaussian Kernel</td>
<td>0.5</td>
</tr>
<tr>
<td>Load</td>
<td>1000</td>
<td>1</td>
<td>0.99</td>
<td>75000</td>
<td>1000</td>
<td>375000</td>
<td>Gaussian Kernel</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Table 4.4: The conditions for the tests conducted using EP

<table>
<thead>
<tr>
<th>Set</th>
<th>Generations</th>
<th>Individuals</th>
<th>Similarity</th>
<th>$\lambda$</th>
<th>$p_{ref}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wind</td>
<td>5000</td>
<td>75</td>
<td>Gaussian Kernel</td>
<td>0.5</td>
<td>0.85</td>
</tr>
<tr>
<td>Load</td>
<td>5000</td>
<td>75</td>
<td>Gaussian Kernel</td>
<td>0.5</td>
<td>0.85</td>
</tr>
</tbody>
</table>

The tests were run at the same time. The results are shown in the next two tables, Table 4.6 and 4.5. Figures 4.24 through 4.27.

Table 4.5: Performance Tests Results - SA

<table>
<thead>
<tr>
<th>Variable</th>
<th>Wind</th>
<th>Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_B$</td>
<td>$5.15 \times 10^{-4}$</td>
<td>$2.01 \times 10^{-4}$</td>
</tr>
<tr>
<td>$IC$</td>
<td>375000</td>
<td>110845</td>
</tr>
<tr>
<td>$STC$</td>
<td>626</td>
<td>735</td>
</tr>
<tr>
<td>$WSC$</td>
<td>20601</td>
<td>75001</td>
</tr>
<tr>
<td>$T$</td>
<td>23.31</td>
<td>331.03</td>
</tr>
<tr>
<td>$p_{ref}$</td>
<td>1.0</td>
<td>2.93 $\times 10^{-4}$</td>
</tr>
<tr>
<td>Run Time (h)</td>
<td>2.019</td>
<td>0.570</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Solution</th>
<th>$H(C^{(1)}\ldots C^{(K)})$</th>
<th>$-15.212$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1^{st}$</td>
<td>$\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>$-922.864$</td>
</tr>
<tr>
<td>$1^{st}$</td>
<td>$O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
<td>$9.564$</td>
</tr>
<tr>
<td>Best</td>
<td>$H(C^{(1)}\ldots C^{(K)})$</td>
<td>$17.204$</td>
</tr>
<tr>
<td>Best</td>
<td>$\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>$-985.021$</td>
</tr>
<tr>
<td>Best</td>
<td>$O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
<td>$13.527$</td>
</tr>
</tbody>
</table>
4.4 Study Case 3: Scenario Reduction

Table 4.6: Performance Tests Results - EP

<table>
<thead>
<tr>
<th>Results using EP</th>
<th>Wind</th>
<th>Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run Time (h)</td>
<td>2.033</td>
<td>1.579</td>
</tr>
<tr>
<td>Best Initial Solution</td>
<td>$H(C^{(1)} \ldots C^{(K)})$</td>
<td>14.324</td>
</tr>
<tr>
<td>Best Initial Solution</td>
<td>$\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-987.855</td>
</tr>
<tr>
<td>Best Initial Solution</td>
<td>$O(C^{(1)} \ldots C^{(K)} ; \lambda)$</td>
<td>12.101</td>
</tr>
<tr>
<td>Best Solution</td>
<td>$H(C^{(1)} \ldots C^{(K)})$</td>
<td>18.812</td>
</tr>
<tr>
<td>Best Solution</td>
<td>$\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-983.42</td>
</tr>
<tr>
<td>Best Solution</td>
<td>$O(C^{(1)} \ldots C^{(K)} ; \lambda)$</td>
<td>14.323</td>
</tr>
</tbody>
</table>

Figure 4.24: Objective Function behaviour during performance test with SA for wind scenarios

Figure 4.25: Objective Function behaviour during performance test with EP for wind scenarios

Figure 4.26: Objective Function behaviour during performance test with SA for load scenarios

Figure 4.27: Objective Function behaviour during performance test with EP for load scenarios
The number of generations for EP and the number of maximum iterations ($IC^{MAX}$) for SA were chosen so that the total number of solutions created in both methods were the same, 375000. From the results, for both scenario sets EP performs better despite in both cases EP had a longer run time. For both scenarios EP generated a higher initial solution, this is happens because EP is a population based method, initially 75 Individuals, i.e. Solutions, were created against the one solution that SA starts from.

Taking into account the obtained results and the stopping criteria that were used, a few remarks are noted about the stopping criterion. In SA the algorithm stopped in two different situations, with the wind scenario set it stopped when it reached $IC^{MAX}$ but with the load scenario set, it stopped when it reached $WSC^{MAX}$. In the first case it’s noticeable a slower evolution which is responsible for the “activation” of that stopping criterion and better results could still be found, whilst in the second test features a fast evolution and then the objective function stalls in the next 75000 iterations, meaning that better results were not found in that period. EP features a fixed stopping criterion meaning that when slow evolution occurs the algorithm may stop in value that is not optimal. To ensure that the maximum value is found, one can set the stopping criteria variables to higher values at the cost of having higher running times. The tests were run in a personal computer with an Intel i5-2400 processor rated at 3.1 GHz and 8GB of RAM.

Considering the results of these performance tests, EP was chosen for the next batch of tests.

4.4.2 Effects of the weight $\lambda$

To understand the effects that the weight $\lambda$ has on the scenario reduction, 11 tests with values for $\lambda$ ranging from 0 to 1 with an interval of 0.1 were conducted. The tests were conducted in the conditions in Table 4.7.

<table>
<thead>
<tr>
<th>Testing conditions</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$Set$</td>
<td>Generations</td>
<td>Individuals</td>
<td>Similarity</td>
<td>$p_{ref}$</td>
</tr>
<tr>
<td>Wind</td>
<td>10000</td>
<td>100</td>
<td>Gaussian Kernel</td>
<td>0.85</td>
</tr>
</tbody>
</table>

After running the tests in the stated conditions, the results were treated and are shown in Tables 4.8 through 4.10.
Table 4.8: Results for values of the weight $\lambda$ between 0 and 0.3

<table>
<thead>
<tr>
<th>Variable</th>
<th>$\lambda = 0$</th>
<th>$\lambda = 0.1$</th>
<th>$\lambda = 0.2$</th>
<th>$\lambda = 0.3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Initial Sol. $H(C(1)\ldots C(K))$</td>
<td>-11.042</td>
<td>-11.991</td>
<td>-12.572</td>
<td>-11.888</td>
</tr>
<tr>
<td>Best Initial Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1411.256</td>
<td>-1412.326</td>
<td>-1413.389</td>
<td>-1414.256</td>
</tr>
<tr>
<td>Best Initial Sol. $O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
<td>-11.0422</td>
<td>-9.379</td>
<td>-7.231</td>
<td>-4.079</td>
</tr>
<tr>
<td>Best Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1410.246</td>
<td>-1410.156</td>
<td>-1411.457</td>
<td>-1410.286</td>
</tr>
<tr>
<td>Best Sol. $O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
<td>-8.441</td>
<td>-6.169</td>
<td>-4.863</td>
<td>-1.634</td>
</tr>
</tbody>
</table>

Table 4.9: Results for values of the weight $\lambda$ between 0.4 and 0.7

<table>
<thead>
<tr>
<th>Variable</th>
<th>$\lambda = 0.4$</th>
<th>$\lambda = 0.5$</th>
<th>$\lambda = 0.6$</th>
<th>$\lambda = 0.7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Initial Sol. $H(C(1)\ldots C(K))$</td>
<td>-11.848</td>
<td>-11.734</td>
<td>-11.895</td>
<td>-12.365</td>
</tr>
<tr>
<td>Best Initial Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1413.151</td>
<td>-1412.431</td>
<td>-1412.373</td>
<td>-1412.911</td>
</tr>
<tr>
<td>Best Initial Sol. $O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
<td>-1.456</td>
<td>1.195</td>
<td>3.716</td>
<td>6.181</td>
</tr>
<tr>
<td>Best Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1410.421</td>
<td>-1410.738</td>
<td>-1410.688</td>
<td>-1410.847</td>
</tr>
<tr>
<td>Best Sol. $O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
<td>-0.281</td>
<td>2.622</td>
<td>4.938</td>
<td>7.170</td>
</tr>
</tbody>
</table>

Table 4.10: Results for values of the weight $\lambda$ between 0.8 and 1.0

<table>
<thead>
<tr>
<th>Variable</th>
<th>$\lambda = 0.8$</th>
<th>$\lambda = 0.9$</th>
<th>$\lambda = 1.0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Initial Sol. $H(C(1)\ldots C(K))$</td>
<td>-11.041</td>
<td>-11.418</td>
<td>-21.171</td>
</tr>
<tr>
<td>Best Initial Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1411.850</td>
<td>-1413.021</td>
<td>-1421.76</td>
</tr>
<tr>
<td>Best Initial Sol. $O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
<td>9.087</td>
<td>11.575</td>
<td>14.218</td>
</tr>
<tr>
<td>Best Sol. $H(C(1)\ldots C(K))$</td>
<td>-8.537</td>
<td>-8.174</td>
<td>-26.060</td>
</tr>
<tr>
<td>Best Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1411.116</td>
<td>-1410.070</td>
<td>-1427.36</td>
</tr>
<tr>
<td>Best Sol. $O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
<td>9.582</td>
<td>11.873</td>
<td>14.274</td>
</tr>
</tbody>
</table>

Since that the desired behaviour is to maximize the value of $H(C^{(1)}\ldots C^{(K)})$ and minimize $\sum_{i=1}^{K} h(C^{(i)})$, it’s expected that in the results one could see a meeting point between the behaviour expressed for $\lambda = 0$ and $\lambda = 1$. Seeing that the value for $\lambda$ exists somewhere between 0.9 and 1.0, new tests were run varying $\lambda$ in this interval. Table 4.11.
Table 4.11: Results for values of the weight $\lambda$ between 0.985 and 0.995

<table>
<thead>
<tr>
<th>Variable</th>
<th>$\lambda = 0.985$</th>
<th>$\lambda = 0.990$</th>
<th>$\lambda = 0.995$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Initial Sol. $H(C^{(1)} \ldots C^{(K)})$</td>
<td>-11.442</td>
<td>-13.838</td>
<td>-20.164</td>
</tr>
<tr>
<td>Best Initial Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1412.180</td>
<td>-1415.703</td>
<td>-1421.001</td>
</tr>
<tr>
<td>Best Initial Sol. $O(C^{(1)} \ldots C^{(K)}, \lambda)$</td>
<td>13.738</td>
<td>13.877</td>
<td>14.038</td>
</tr>
<tr>
<td>Best Sol. $H(C^{(1)} \ldots C^{(K)})$</td>
<td>-11.035</td>
<td>-11.942</td>
<td>-24.784</td>
</tr>
<tr>
<td>Best Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1415.438</td>
<td>-1416.187</td>
<td>-1426.366</td>
</tr>
<tr>
<td>Best Sol. $O(C^{(1)} \ldots C^{(K)}, \lambda)$</td>
<td>13.777</td>
<td>13.901</td>
<td>14.068</td>
</tr>
</tbody>
</table>

For any value below 0.985, the algorithm will only maximize the value for $H(C^{(1)} \ldots C^{(K)})$ and for any value above 0.995, the algorithm will only minimize the value of $\sum_{i=1}^{K} h(C^{(i)})$. So it’s possible to conclude that the optimal value for $\lambda$ lies between 0.985 and 0.995, with 0.99 having the desired behaviour. This value was used for further testing.

4.4.3 Scenario Reduction

To reduce the number of scenarios, the conditions are specified as:

- Number of representative scenarios: 25
- Number of Generations: 10000
- Individuals: 150
- Probability Reference: 0.85
- $\lambda = 0.99$
- Gaussian Kernel Similarity

4.4.3.1 Load Scenario Reduction

The results are presented in Table 4.12 and Figures 4.28 through 4.31.

Table 4.12: Results for reduction from 193 to 25 load forecast scenarios

<table>
<thead>
<tr>
<th>Variable</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Initial Sol. $H(C^{(1)} \ldots C^{(K)})$</td>
<td>-19.1013</td>
</tr>
<tr>
<td>Best Initial Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1422.0047</td>
</tr>
<tr>
<td>Best Initial Sol. $O(C^{(1)} \ldots C^{(K)}, \lambda)$</td>
<td>13.8868</td>
</tr>
<tr>
<td>Best Sol. $H(C^{(1)} \ldots C^{(K)})$</td>
<td>-16.6632</td>
</tr>
<tr>
<td>Best Sol. $\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1421.0638</td>
</tr>
<tr>
<td>Best Sol. $O(C^{(1)} \ldots C^{(K)}, \lambda)$</td>
<td>13.9019</td>
</tr>
</tbody>
</table>
4.4 Study Case 3: Scenario Reduction

Figure 4.28: Objective Function behaviour during the load scenario reduction

Figure 4.29: Identified load representative scenarios
Results in Study Cases

Figure 4.30: Clusters and representative scenario - clusters 1 to 16

Figure 4.31: Clusters and representative scenario - clusters 17 to 25
Regarding the results in Table 4.12, the algorithm may have not behaved as it was wanted but the criterion \( \sum_{i=1}^{K} h(C(i)) \) had a value low enough that the desirable result of existing clusters with only one element is satisfactory. The representative scenarios to be used for stochastic programming are shown in Figure 4.29. Regarding the formed clusters, some significant outliers are represented by its own scenario, forming clusters of a single scenario as it can be seen in Figure 4.30 in positions (1,4), (2,2), (2,4) and (3,4), in which \((i, j)\), \(i\) represents the row and \(j\) represents the column of figures.

Although there are clusters that are comprised of very similar scenarios, there are some that are not the case. Cluster in position (3,3) in Figure 4.31, is comprised of very different scenarios and treating the members of this cluster separately rather by representing them by a single scenario is recommended. For instance, the scenario reduction procedure could be applied to only that cluster to separate the more strange scenarios.

### 4.4.3.2 Wind Scenario Reduction

The results are presented in Table 4.13 and Figures 4.32 through 4.35.

<table>
<thead>
<tr>
<th>Wind Scenario Reduction</th>
<th>Variable</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Initial Sol.</td>
<td>( H(C^{(1)}\ldots C^{(K)}) )</td>
<td>15.0394</td>
</tr>
<tr>
<td>Best Initial Sol.</td>
<td>( \sum_{i=1}^{K} h(C(i)) )</td>
<td>-987.2284</td>
</tr>
<tr>
<td>Best Initial Sol.</td>
<td>( O(C^{(1)}\ldots C^{(K)}, \lambda) )</td>
<td>9.9240</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>( H(C^{(1)}\ldots C^{(K)}) )</td>
<td>13.5286</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>( \sum_{i=1}^{K} h(C(i)) )</td>
<td>-989.1601</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>( O(C^{(1)}\ldots C^{(K)}, \lambda) )</td>
<td>9.9280</td>
</tr>
</tbody>
</table>

Figure 4.32: Objective Function behaviour during the wind scenario reduction
Figure 4.33: Identified wind power representative scenarios

Figure 4.34: Clusters and representative scenario - clusters 1 to 16
4.4 Study Case 3: Scenario Reduction

Regarding the reduction process in wind power scenarios, seemingly no significant outlier scenarios have been identified. Analysing the clusters in figures 4.34 and 4.35, the scenarios seem to be well clustered. It can also be seen in the results in Table 4.13 that the algorithm gave more focus minimizing $\sum_{i=1}^{K} h(C^{(i)})$ and that resulted in decreasing the value of $H(C^{(1)} \ldots C^{(K)})$. For reducing the wind power scenarios, it should found the value of $\lambda$ that creates the desired behaviour.

4.4.4 Different Similarity Measures

The previous tests were conducted using the Gaussian Kernel similarity measure explained in Section 3.3.1. To understand how the similarity measure influences the clustering of the scenarios, new tests were conducted with the other similarity measures explained in Section 3.3, Cosine similarity and Kohonen similarity.

The conditions for these tests were:

- Number of representative scenarios: 25
- Number of Generations: 10000
- Individuals: 100
- Probability Reference: 0.85
- $\lambda = 0.99$
4.4.4.1 Cosine Similarity

Applying the Cosine similarity to load forecast scenario reduction, the results are shown in Table 4.14 and Figures 4.36 to 4.36

Table 4.14: Results obtained using Cosine similarity in load forecast scenario reduction

<table>
<thead>
<tr>
<th>Variable</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Initial Sol.</td>
<td>$H(C^{(1)}\ldots C^{(K)})$</td>
</tr>
<tr>
<td>Best Initial Sol.</td>
<td>$\sum_{i=1}^{K} h(C^{(i)})$</td>
</tr>
<tr>
<td>Best Initial Sol.</td>
<td>$O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>$H(C^{(1)}\ldots C^{(K)})$</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>$\sum_{i=1}^{K} h(C^{(i)})$</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>$O(C^{(1)}\ldots C^{(K)}, \lambda)$</td>
</tr>
</tbody>
</table>

Figure 4.36: Objective Function behaviour with Cosine similarity
Figure 4.37: Identified load representative scenarios with Cosine similarity

Figure 4.38: Clusters and representative scenario with Cosine similarity - clusters 1 to 16
As it was expected, the use of a different similarity measure produced different clusters than the ones presented in the previous tests. From Table 4.14 the algorithm behaved differently, with a minimization of the two values. The specified value $\lambda$ does not apply for this case, and a lower value is recommended. The identified outliers and representative scenarios are different from what was obtained with the Gaussian Kernel similarity. There are scenarios that are well clustered but some clusters like the ones in position (2,4) and in position (4,2) in Figure 4.38 need to be treated differently and further reduction is recommended.

### 4.4.4.2 Kohonen Similarity

For this similarity measure, the results are shown in Table 4.15 and Figures 4.40 to 4.40.

Table 4.15: Results obtained using Kohonen similarity in load forecast scenario reduction

<table>
<thead>
<tr>
<th>Kohonen Similarity Results</th>
<th>Variable</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Initial Sol.</td>
<td>$H(C^{(1)} \ldots C^{(K)})$</td>
<td>-17.800</td>
</tr>
<tr>
<td>Best Initial Sol.</td>
<td>$\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1419.735</td>
</tr>
<tr>
<td>Best Initial Sol.</td>
<td>$O(C^{(1)} \ldots C^{(K)}, \lambda)$</td>
<td>13.877</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>$H(C^{(1)} \ldots C^{(K)})$</td>
<td>-16.310</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>$\sum_{i=1}^{K} h(C^{(i)})$</td>
<td>-1420.579</td>
</tr>
<tr>
<td>Best Sol.</td>
<td>$O(C^{(1)} \ldots C^{(K)}, \lambda)$</td>
<td>13.901</td>
</tr>
</tbody>
</table>
4.4 Study Case 3: Scenario Reduction

Figure 4.40: Objective Function behaviour with Kohonen similarity

Figure 4.41: Identified load representative scenarios with Kohonen similarity
Figure 4.42: Clusters and representative scenario with Kohonen similarity - clusters 1 to 16

Figure 4.43: Clusters and representative scenario with Kohonen similarity - clusters 17 to 25
As it happened with the change to the Cosine similarity, with $\lambda = 0.99$ the algorithm did not behaved as intended as both criteria were minimized. The results of this effect are seen in Figures 4.42 and 4.43 with several clusters with one or two scenarios and a few clusters like the ones in positions (2,4), (4,1) in Figure 4.42 and in position (2,3) in Figure 4.43 that have a few scenarios that are not so similar.

Concluding, the use of different similarity metrics implies that a different value for $\lambda$ must be implemented. The "weight" $\lambda$ must be carefully chosen for each situation so that the algorithm behaves like it is intended.
Chapter 5

Conclusions and Future Work

This last Chapter the main conclusions that were made from the scenario reduction tests that were conducted in Chapter 4. The second section proposes a few recommendations of what it could be done to continue this work.

5.1 Conclusions

The main objective for this Thesis was to propose a methodology to identify strange but potentially relevant scenarios in a scenario reduction process. The methodology was formulated around an optimization problem to maximize the entropy between clusters and minimizing the sum of entropies of each clusters.

First, to understand the underlying concepts in the proposed methodology, PDF estimation based in the Parzen Windows technique and the ITMS algorithm were studied and functions were developed in Matlab. Both functions function as desired and the expected results were achieved proving the usefulness of these tools. Study case 1 shows the representation of the PDF of a dataset estimated using the Parzen Windows technique. In study case 2, the ITMS successfully found the modes the analysed datasets’ PDF and a comparison between the results of ITMS and the respective PDF estimated in study case 1 shows ITMS great capabilities in mode finding.

To solve the optimization problem, SA and EP were used. EP achieved better results for the same conditions although SA is a faster algorithm. This is due to computational reasons, as EP is a population based method which involves more calculations per generation in contrast to SA that only features the creation of a single neighbour solution per generation. The fact that more solutions are evaluated and selected in EP gives this metaheuristic more chances to find the optimal solution. Also, the fact that several solutions are produced in the initial instances of the algorithm, may give a better starting point for EP than it does for SA.

To control the behaviour of the optimization algorithm two parameters, a "weight" $\lambda$ and a scale factor were introduced. To achieve the desired behaviour, a maximization of the entropy between clusters and a minimization of the entropy in each cluster, $\lambda$ must be specified to each situation, in the conducted tests with load scenarios $0.985 \leq \lambda \leq 0.995$. The following tests
showed that this interval may not be true to other similarity measures or other kind of scenario sets, such as wind forecast scenarios.

The choice of a given similarity measure must be done according to the nature of the problem. The main similarity measure used was the Gaussian kernel in which the clusters were formed by exploiting the presented Entropy concepts. The other two similarity measures produced similar clusters, but as the parameter $\lambda$ had the value for the previous optimization, the algorithm’s behaviour was different from what was desired.

Due to time restrictions during the development of this work the calculation of the probabilities for each representative scenario as not analysed.

Remembering, the representative scenarios identified here are to be used to represent the uncertainty of a forecast in stochastic programming to solve an unit commitment problem instead of using the full set of scenarios. The use of reduced set of scenarios reduces considerably the computational time required by the program, opening the possibility to use stochastic programs to solve unit commitment problems.

Concluding, the proposed methodology does identify relevant scenarios but the parameter $\lambda$ and the similarity measure must be chosen in accordance to the problem. To produce better results, more generations or more individuals could be used but this would require higher computational times.

5.2 Future Work

Although the main objective of representing the strange scenarios has been achieved, there is still the possibility to pursue this work. Taking this into account, a few recommendations for future are:

- As there are several scenario reduction techniques already published, a comparison between the proposed technique in this Dissertation and another established scenario reduction technique could be done.

- It’s encouraged to measure the difference between an estimated PDF with scenarios generated using a Monte Carlo process and the one estimated using only the representative scenario set.

- Exploring other forms to choose a representative scenario of a cluster and note the differences between the different methods that result from a stochastic program to solve a unit commitment problem.

- This work focused in finding the potentially relevant scenarios with low representation, but as the number of final scenarios is fixed some outliers could still be clustered with the other scenarios. One could, for example, reapply the scenario reduction procedure to clusters with higher internal entropy to root out outliers. A successive distillation of clusters with very different scenarios is proposed.
5.2 Future Work

- Run a stochastic program for solving an unit commitment problem with the resulting representative scenarios and their probabilities and compare it to a solution with all the scenarios.

- It’s proposed to verify if the solution created from stochastic model using the reduced set of scenarios survives the outlier scenario.

- Use other optimization tools or metaheuristics such as EPSO to improve the results.

- Use different similarity metrics.

Lastly, a suggestion for a research question: Does an estimated PDF with Gaussian Kernels and representative scenarios approximates to a PDF estimated using a Monte Carlo process? It’s hypothesized that the answer is yes, but since there was no time during the development of this Thesis to prove this statement, further research into the matter is encouraged.
Appendix A

ITMS Results

A.1 Dataset 1

Figure A.1: Dataset 1 - Test 1 results

Figure A.2: Dataset 1 - Test 2 results

Figure A.3: Dataset 1 - Test 3 results

Figure A.4: Dataset 1 - Test 4 results
A.2 Dataset 2

Figure A.11: Dataset 1 - Test 11 results

Figure A.12: Dataset 1 - Test 2 results

Figure A.13: Dataset 2 - Test 1 results

Figure A.14: Dataset 2 - Test 2 results

Figure A.15: Dataset 2 - Test 3 results

Figure A.16: Dataset 2 - Test 4 results
A.3 Dataset 3

Figure A.23: Dataset 2 - Test 11 results
Figure A.24: Dataset 2 - Test 12 results

Figure A.25: Dataset 3 - Test 1 results
Figure A.26: Dataset 3 - Test 3 results

Figure A.27: Dataset 3 - Test 4 results
Figure A.28: Dataset 3 - Test 5 results
Figure A.29: Dataset 3 - Test 6 results

Figure A.30: Dataset 3 - Test 7 results

Figure A.31: Dataset 3 - Test 8 results

Figure A.32: Dataset 3 - Test 9 results

Figure A.33: Dataset 3 - Test 10 results

Figure A.34: Dataset 3 - Test 11 results
Figure A.35: Dataset 3 - Test 12 results
Appendix B

Article for submission

The article detailing the relevant work developed in this Thesis is annexed here. The article will be submitted for a journal or conference for Power Systems.
Finding Representative Scenarios in Wind Power or Load Forecast

Tiago Rodrigues, Jean Sumaili, Member, IEEE, and Vladimiro Miranda, Fellow, IEEE

Abstract—This paper proposes a load or wind power forecast scenario reduction methodology using a clustering technique based on an optimization of Entropy criteria capable of still representing the uncertainty in a given forecast. Previous works in this field neglect the identification of scenarios with low occurrence, i.e., outliers, that can cause significant consequences in case they occur. The proposed methodology clusters a large set of forecast scenarios in smaller sets based on their similarity and still keeps identified the more strange outlier scenarios. The clusters of scenarios are then represented by a focal scenario. The use of reduced set of scenarios opens the possibility to solve Unit Commitment problems using stochastic programming by reducing its the computational burden. The representation of the relevant outlier scenarios gives decision makers more insight when working with reduced sets of scenarios.

Index Terms—clustering, entropy, forecasting, load demand, metaheuristics, scenario reduction, similarity, uncertainty, wind power.

I. INTRODUCTION

The need of a large set of scenarios is needed to accurately represent the uncertainty in a given forecast, load or wind power, has hindered the use of multi-stage stochastic programming models to solve the Unit Commitment (UC) problem due to the implied heavy computational burden. Using these tools, one assumes that the uncertainty evolves as a discrete time continuous stochastic process. The need for scenario reduction has prompted the research of several methods to reduce the number of scenarios needed to represent the uncertainty present in the original set of forecast scenarios.

The most common scenarios reduction techniques are based in scenario tree construction that were first proposed in [1] and its application to power management problems in [2]. A methodology for wind power scenario reduction is done in [3] and in [4] they apply the same concepts to risk-averse electricity trading. Clustering techniques are used in [5] and [6] to reduce the number of wind power scenarios in a large set produced by a Monte Carlo sampling process from the Probability Density Function (PDF) representing the historical error distribution of a wind forecast.

During a scenario reduction process, potentially relevant outlier scenarios may be excluded from the analysis of the problem and in case they occur there could be significant consequences. The main motivation for this work is to give decision makers more insight on the consequences of working with a reduced set of scenarios, be it with load or wind forecast. The proposed methodology features a similarity based clustering technique to reduce the number of scenarios of wind power or load forecast and maintaining a representation of the potentially relevant outlier scenarios. The clustering technique is formulated as an optimization problem to create a fixed number of clusters of scenarios that maximize the Entropy between clusters and minimizes the sum of the entropies of all the clusters. To solve the optimization problem, the metaheuristics Simulated Annealing (SA) and Evolutionary Programming (EP) were used.

II. METHODOLOGY

The objective of the proposed methodology is: find, around a fixed number of representative scenarios, the scenarios that create the same number of clusters while maximizing the Entropy between the clusters and minimizing the sum of the entropies in each cluster. The Entropy between clusters and the entropy in a clusters are Cluster Evaluation Functions (CEF) that were proposed in [7] based in the entropy concepts introduced by Renyi and Shannon in [8] and [9] that measure the divergence between clusters and in a cluster, respectively. Maximizing the first CEF will result in a set of clusters that are the most different from each other and minimizing the second will create clusters that group the most similar ones and leaving the ones that can’t be clustered in its own cluster, the Entropy in a cluster is lowest when it has only one scenario. The scenarios are clustered according a similarity measure and the optimization problem is solved mainly using EP as it proved to be the method that produced the better results.

A. Scenario and Cluster Definitions

The forecast scenarios used in this work were provided by Dr. Sumaili. The wind power scenarios were obtained as in [5] and have a dimension of 24 points, one for each hour, while the load scenarios are defined as in [10] and have a dimension of 96 points using 15-min forecast data. Considering a dataset with $M$ scenarios in a time horizon $T$ such that, $x^{(m)} = \{x_{1}^{(m)}, \ldots, x_{T}^{(m)}\}$, the reduction procedure creates, for specified $K$ number of representative scenarios, $K$ clusters so that $C^{(k)} \subset X$, for $k = 1, \ldots, K$ and $1 \leq K \leq M$ and verifying $N^{(1)} + \ldots + N^{(K)} = M$, with $N^{(k)}$ being the number of scenarios in $C^{(k)}$.

B. Objective Function

The clustering technique is based in an optimization problem which is formulated in Equation 1.

$$\max \{O(C^{(1)}, \ldots, C^{(K)}), \lambda\} =$$

$$= (1 - \lambda) \times H(C^{(1)}, \ldots, C^{(K)}) - \lambda \times \sum_{i=1}^{K} h(C^{(i)}) \quad (1)$$
\[ H(C_1, \ldots, C_k) = -\ln V(C_1, \ldots, C_k) \]
\[ V(C_1, \ldots, C_k) = \frac{1}{2 \prod_{k=1}^{K} N_k^{M} \sum_{i=1}^{M} \sum_{j=1}^{M} G(x_i - x_j, \Sigma_x)} \tag{2} \]
\[ h(C_k) = -\ln \sum_{i=1}^{N_k} \sum_{j=1}^{N_k} G(x_i - x_j, \Sigma_x) \tag{3} \]
\[ G(x_i - x_j, \Sigma_x) = \frac{1}{\sqrt{(2\pi)^n \det(\Sigma_x)}} \exp \left( \frac{-1}{2}(x_i - y_j)\Sigma_x^{-1}(x_i - x_j)^T \right) \tag{4} \]

\[ H(C^{(1)}, \ldots, C^{(K)}) \] represents the entropy between clusters as shown in equation 2. \( h(C^{(i)}) \) represents the entropy of a cluster \( i \) calculated as in Renyi’s quadratic Entropy in Equation 3. As this is a multi-criterion optimization a “weight” \( \lambda \) was introduced so that when \( \lambda = 0 \), the objective function only maximizes the entropy between clusters and when \( \lambda = 1 \), the sum of the entropies of each cluster is minimized. \( G \) represents the Gaussian Kernel function where \( \Sigma_x \) is the \( T \times T \) diagonal covariance matrix measured for the dataset.

\section*{C. Similarity Measures}

The main similarity measure used in this work was a metric evaluated in a non-Euclidean space according to the entropy principles to define the similarity between two scenarios as in [10]. Equation 5 shows the similarity measure using the Gaussian kernel in Equation 4.

\[ s_{x_i, y_j} = G(x_i - y_j, \Sigma_x) \tag{5} \]

Other similarity measures were tested such as the Cosine Similarity and the Kohonen Similarity which are defined in [11].

\section*{D. Evolutionary Programming}

EP is part of the Evolutionary Computation algorithms which take cues from biological evolution paradigms, most notably Darwin’s theory of evolution. EP is a population based algorithm proposed by Fogel et al in 1966 in [12]. In this algorithm, an initial population of individuals, i.e. solutions, is created and evaluated calculating the objective function in 1 and in each generation the population is cloned and mutated to generate new solutions. The new mutated population is evaluated too. Then the individuals to be carried to the next generation are selected through a stochastic tourney.

In the classical EP algorithms, the mutation scheme is done by adding small mutations to the variables of an individual. In this work, as the variables of an individual are the scenarios that aggregates the clusters, applying mutations to a curve would lead to very irregular curves that would serve as aggregators. A new “mutation” scheme was devised to select a new aggregator scenario for each cluster based in the former aggregator scenarios. An \( M \times M \) similarity matrix \( S \) is created for the set of scenarios and to aid the choice of new scenarios in the mutation process, a probability distribution (PD) is created to weight the probability of a scenario being chosen based on the similarity results in Equation 6 for a scenario \( i \). A vector containing all of the \( p(S(i, j)) \) is then sorted in a descending order. A cumulative distribution function (CDF) is created as in Equation 7 and favours the selection of more similar scenarios.

\[ p(S(i, j)) = \frac{S(i, j)}{\sum_{j=1}^{M} S(i, j)} \text{, for } j = 1, \ldots, M \tag{6} \]

\[ c_k(S(i, j)) = \sum_{k=1}^{M} p(S(i, j)) \text{ for } k = 1, \ldots, M \tag{7} \]

The mutation process involves the selection of a new scenario for each cluster and for each individual by drawing a random probability and comparing it to the CDF of the former scenario. The closest CDF value to the probability determines the next scenario. After all the new individuals are created, the selection process begins by finding the best individual between the two populations and passing it to next generation. The rest individuals are submitted to a stochastic tourney in which two randomly selected individuals are compared and a probability is randomly generated. If the probability is inferior to a probability reference, \( P_{ref} \), then the better one carries on and if not, the other carries on. This allows the algorithm to have a rising evolution and not stop at local maximums as would have occurred if an elitist selection process was implemented.

The SA algorithm features neighbour selection process identical to the one explained here. The SA detailed in [13] served as the main template.

\section*{E. Representative Scenario}

This methodology uses the aggregator scenario to represent each clusters. Papers [5] and [6] detail other possibilities for the selection of a representative scenario and detail the implications of this choice.

\section*{III. Results}

\begin{figure}[h!]
\centering
\includegraphics[width=\textwidth]{load_power_forecast.png}
\caption{Load (left) and Wind Power (right) forecast scenarios}
\end{figure}

The initial sets of scenarios were provided by Dr. Sumaili. The load and wind power forecast scenarios are shown in Figure 1. Preliminary tests showed that both criteria had values that are very far apart, for example in one test, \( H(C^{(1)}, \ldots, C^{(K)}) = -20.03 \) and \( \sum_{i=1}^{K} h(C^{(i)}) = -1419.96 \). This difference in values could cause the algorithm to give more focus to solutions that minimize the second criterion and to counter this effect the second criterion was divided
by scale factor of 100 and λ tuned so that the objective function evolves by increasing the value of $H(C^{(1)},...,C^{(K)})$ and decreasing the value of $\sum_{i=1}^{K} h(C^{(i)})$. Performance tests showed that the EP algorithm achieved better results than SA, but the latter was faster. Further testing showed that for $0.985 \leq \lambda \leq 0.995$ search the algorithm maximizes $H(C^{(1)},...,C^{(K)})$ and minimizes $\sum_{i=1}^{K} h(C^{(i)})$.

The final reduction tests were done with the following conditions: 25 representative scenarios; 10000 generations; 150 individuals per population; probability reference of 0.85; $\lambda = 0.85$; Gaussian Kernel Similarity. The results are shown in Figure 2 and 3.

![Fig. 2. Clusters of scenarios with their representative for load scenarios](image1)

![Fig. 3. Clusters of scenarios with their representative for wind scenarios](image2)

From the load results, it’s possible to see that the algorithm behaved as intended. The scenarios were well clustered and important outlier scenarios are well identified. However scenarios in the 4th row in columns 1 and 2 require special attention and further applying the reduction procedure to these clusters is recommended. With the wind power scenarios, the results weren’t so favourable. Although the scenarios are well clustered, no relevant outlier scenarios were identified. This may be due to the value of λ that could be different for this set of scenarios, an slightly higher value is recommended to identify the outlier scenarios.

**IV. Conclusion**

The methodology successfully manages to cluster the similar scenarios and identify the relevant outlier scenarios, opening the possibility of use of the stochastic programming models to solve UC problems by using the identified representative scenarios. EP was the preferred metaheuristic to solve the optimization problem as it performed better than SA given the fact that the former is a population based method, but any optimization tool can be used.

The choice of the similarity metric must be done in accordance to the problem and value for λ must be found so that the objective function maximizes the Entropy between clusters and minimizes the Entropy in each cluster at the same time, both criteria tend to evolve in the same direction. Due to time restrictions the probabilities for each scenarios could not be calculated.

The reduced set of scenarios should be used to solve a stochastic UC problem and the identified outlier scenarios could be used to see if solution is still valid and survives the appearance of any of the outlier scenarios. It’s also encouraged to verify if the estimated PDF using Gaussian Kernels and the representative scenarios approximates to a PDF estimated using a Monte Carlo process.

**REFERENCES**


References


REFERENCES


