Intelligent Methods for
Pattern Recognition and Optimisation

by

Nedyalko Petrov

School of Computing

University of Portsmouth, UK

The thesis is submitted in partial fulfilment of the requirements for the award of the degree of Doctor of Philosophy of the University of Portsmouth

January 2015
Copyright

Copyright © 2015 Nedyalko Petrov. All rights reserved.

The copyright of this thesis rests with the Author. Copies (by any means) either in full, or of extracts, may not be made without prior written consent from the Author.
Declaration

The work presented in this thesis has been carried out in the School of Computing at the University of Portsmouth under the supervision of Dr. Ivan Jordanov.

Whilst registered as a candidate for the above degree, I have not been registered for any other research award. The results and conclusions embodied in this thesis are the work of the named candidate and have not been submitted for any other academic award.

Portsmouth,

Nedyalko Petrov
To All That Is. To Myself.
Abstract

This dissertation presents and discusses the processes of investigation, implementation, testing, validation and evaluation of several computational intelligence-based systems for solving four large-scale real-world problems. In particular, two industrial problems from the pattern recognition and two from the process optimisation areas are studied and intelligent methods to address them are proposed, developed and tested using real-world data.

The first problem investigated is the application of an intelligent visual inspection system for classification of texture images. Two major approaches, incorporating supervised and unsupervised (without a priori knowledge) learning techniques, are considered and neural network based classifiers are trained. The focus is kept on the application of unsupervised non-linear dimensionality reduction techniques in combination with unsupervised classification methods. A number of experiments and simulations are performed to evaluate the proposed approaches and the results are critically compared.

Next, a classification problem for timely and reliable identification of emitters of radar signals is investigated. A large data set, containing a considerable amount of missing data is used. Several techniques for dealing with the incomplete data values are employed, including listwise deletion and multiple imputation. Methods incorporating neural network classifiers are studied and the proposed approaches are tested and validated over a number of simulations in the MATLAB environment.

The third large-scale problem, presented in this work, addresses the need for optimisation of a thermodynamics first principle-based prediction model for simulation of a major purifying process, used in British Petroleum (BP)
refineries. A technique incorporating genetic algorithms is applied for optimising a number of the model parameters and for closing up the gaps between the predicted and measured data. Several functions and a graphical user interface (GUI) tool are implemented in MATLAB to assist the analysis, optimisation, testing and validation of the investigated model. Significant overall improvement in its prediction capabilities is achieved.

The final problem, covered in this research work, is the need to improve the convergence rate of a computationally very expensive aerodynamic optimisation process. It is addressed by exploring some physics-grounded heuristics and presenting a novel intelligent approach for automated shape optimisation. A set of basis functions (for spanning the design space) is derived in such a way that they facilitate the work of a time-consuming and expensive computational fluid dynamics (CFD) optimisation process. Two MATLAB-based GUI tools are developed to support the calculation, exploration, testing and validation of the studied approach. Experiments for optimising real aircraft geometry are run on supercomputers through an industrial partner (AIRBUS Operations Ltd). The initial results show very promising opportunities for improving the convergence rate of the slow optimisation process.
Acknowledgements

I would like to express my sincerest and deepest gratitude to my supervisor Dr. Ivan Jordanov for his continuous support, motivation, competent guidance and friendship. Without his encouragement and positive attitude towards my efforts, the work presented in this dissertation would not have been accomplished. I would also like to thank his wonderful wife Rossi for the enjoyable time spent during our trips to different congresses and conferences and for her kindness and support.

The experience of this research endeavour would not have been as exciting, unique and motivating without all the moments shared with my peers, collaborators and colleagues at the University of Portsmouth and during my placements with BP and Airbus. I would like to thank them all for making it such a memorable and enjoyable experience.

My thanks go also to the School of Computing at the University of Portsmouth for the provided sponsorship and pleasant working environment during my research.

I would also like to acknowledge the financial support received from the Engineering and Physical Sciences Research Council (EPSRC) and Industrial Mathematics Knowledge Transfer Network (IM KTN) that enabled the conduction of two short knowledge transfer partnership (sKTP) projects with BP International Ltd and Airbus Operations Ltd.

Last, but not least, my deepest gratitude goes to my parents, Lyuba and Lyubomir, my brother Nikolay, my girlfriend Tsvetelina and the rest of my family for their endless love and support.
# Table of Contents

Copyright .......................................................................................................................... i
Declaration .......................................................................................................................... ii
Abstract .............................................................................................................................. iv
Acknowledgements ............................................................................................................ vi
Table of Contents .............................................................................................................. vii
List of Tables ....................................................................................................................... xi
List of Figures ..................................................................................................................... xiii
Glossary ............................................................................................................................... xvii

## CHAPTER 1 INTRODUCTION ......................................................................................... 1

1.1 MOTIVATION .............................................................................................................. 1
1.2 ORIGINAL CONTRIBUTIONS TO KNOWLEDGE .............................................. 3
1.3 THESIS OUTLINE ................................................................................................. 5

## CHAPTER 2 INTELLIGENT PATTERN RECOGNITION AND CLASSIFICATION FOR TEXTURE IMAGES ......................................................................................... 8

2.1 BACKGROUND KNOWLEDGE ............................................................................... 9
2.2 DATA ACQUISITION AND PREPARATION ......................................................... 15
   2.2.1 Features Extraction ....................................................................................... 16
   2.2.2 Scaling .......................................................................................................... 19
2.3 DIMENSIONALITY REDUCTION ......................................................................... 20
   2.3.1 Principal Component Analysis (PCA) ............................................................. 20
   2.3.2 Nonlinear Principal Component Analysis (NLPCA) ..................................... 21
   2.3.3 Hierarchical NLPCA .................................................................................. 25
   2.3.4 Kernel PCA .................................................................................................. 26
   2.3.5 Linear Discriminant Analysis ..................................................................... 28
2.4 CLASSIFICATION .................................................................................................... 28
2.5 RESULTS .................................................................................................................. 31
2.6 CONCLUSIONS ......................................................................................................................... 43

CHAPTER 3 INTELLIGENT RADAR SIGNAL RECOGNITION AND CLASSIFICATION .... 45

3.1 BACKGROUND KNOWLEDGE ................................................................................................. 46
  3.1.1 Neural Networks in Radar Recognition Systems ............................................................... 51
  3.1.2 Dealing with Missing Data ................................................................................................. 55
3.2 DATA ANALYSIS ..................................................................................................................... 58
3.3 DATA PRE-PROCESSING ......................................................................................................... 60
  3.3.1 Data Cleaning and Imputation ........................................................................................... 60
  3.3.2 Data Coding and Transformation ....................................................................................... 62
  3.3.3 System Training .................................................................................................................. 64
3.4 RESULTS .................................................................................................................................. 64
  3.4.1 Case Study 1 – Listwise Deletion and Feedforward Neural Networks ....................... 65
  3.4.2 Case Study 2 – Multiple Imputation and Feedforward Neural Networks ............... 72
3.5 CONCLUSION .......................................................................................................................... 76

CHAPTER 4 INTELLIGENT OPTIMISATION OF A HYDROTREATING PROCESS ........ 78

4.1 BACKGROUND KNOWLEDGE ................................................................................................. 79
4.2 INVESTIGATED APPROACH .................................................................................................... 81
  4.2.1 Status Quo ......................................................................................................................... 81
  4.2.2 Approach Overview ......................................................................................................... 82
4.3 GAP ANALYSIS ....................................................................................................................... 83
  4.3.1 Acquired Data .................................................................................................................... 83
  4.3.2 Gap Analysis Approach .................................................................................................... 84
  4.3.3 Sulphur Gap Analysis ....................................................................................................... 85
  4.3.4 Nitrogen Gap Analysis ..................................................................................................... 86
4.4 MODEL OPTIMISATION .......................................................................................................... 87
  4.4.1 Linking the Model to MATLAB® ...................................................................................... 87
  4.4.2 GUI Tool for Model Analysis and Optimisation ............................................................... 88
  4.4.3 Optimisation ....................................................................................................................... 89
4.5 OPTIMISATION RESULTS ....................................................................................................... 90
4.6 CONCLUSION ........................................................................................................................... 94
CHAPTER 5 INTELLIGENT PARAMETRIC PATCH PERTURBATION FOR AERODYNAMIC SURFACE OPTIMISATION

5.1 BACKGROUND KNOWLEDGE .............................................................................. 97
5.2 KEY IDEAS ........................................................................................................... 100
5.3 INVESTIGATED APPROACH ............................................................................... 102
  5.3.1 Scenario ......................................................................................................... 102
  5.3.2 Geometry Displacement ................................................................................ 103
  5.3.3 Objective Function ........................................................................................ 104
  5.3.4 Mathematical Definitions .............................................................................. 105
  5.3.5 Streamwise Curvature ................................................................................... 106
  5.3.6 Tensor Product of Basis Functions ............................................................... 106
5.4 DERIVATION OF BASIS FUNCTIONS ................................................................ 107
  5.4.1 Functions with Respect to s and t Directions ................................................ 107
  5.4.2 Continuity of Position, Slope and Curvature ................................................. 108
  5.4.3 Odd and Even Functions .............................................................................. 109
  5.4.4 Cross-stream functions ................................................................................. 109
  5.4.5 Streamwise Functions ................................................................................... 110
5.5 IMPLEMENTATION .............................................................................................. 110
5.6 RESULTS .............................................................................................................. 116
5.7 CONCLUSION ...................................................................................................... 118

CHAPTER 6 CONCLUSION ......................................................................................... 120

6.1 CONCLUSIONS .................................................................................................... 120
6.2 FURTHER DIRECTIONS ....................................................................................... 122
6.3 LIST OF PUBLICATIONS .................................................................................... 123
  6.3.1 Within the Scope of the Dissertation ............................................................ 123
  6.3.2 Outside the Scope of the Dissertation ........................................................... 124

REFERENCES ............................................................................................................. 126

APPENDICES ................................................................................................................ 139

A1. INTELLIGENT PATTERN RECOGNITION AND CLASSIFICATION FOR TEXTURE IMAGES ........ 139
A2. INTELLIGENT RADAR SIGNAL RECOGNITION AND CLASSIFICATION .............................. 144
A3. INTELLIGENT OPTIMISATION OF A HYDROTREATING PROCESS .......................... 150
A4. INTELLIGENT PARAMETRIC PATCH PERTURBATION FOR AERODYNAMIC SURFACE OPTIMISATION ........................................................................................................... 155
A5. WORD CLOUD THESIS SUMMARY ........................................................................ 160
A6. UPR16 FORM .......................................................................................................... 161
### List of Tables

**Table 2-1.** Classification Accuracy Rates (in %) for the Investigated Methods. ............ 37

**Table 3-1.** Letter Nomenclature for Nominal Radar Frequency Bands (IEEE, 1976)...... 48

**Table 3-2.** Commonly Adopted ESM Abbreviations. .................................................. 50

**Table 3-3.** Sample radar data subset. Missing values (i.e. values that could not have been intercepted or recognised) are denoted by ‘–’. The rest of the acronyms are defined in Table 3-4................................................................. 58

**Table 3-4.** Data description and percentage of missing values. In column “Type”: I – integer; C – categorical; R – real values......................................................... 59

**Table 3-5.** Sample radar data subset with imputed values for the missing data entries. .. 62

**Table 3-6.** Sample subset with imputed radar data and natural number coding for the ‘RFC’, ‘PRC’, ‘PDC’, and ‘ST’ signal characteristics. ............................................. 62

**Table 3-7.** Example of binary coding for 32-level categorical variable.......................... 63

**Table 3-8.** Example of dummy coding for 32-level categorical variable. ...................... 63

**Table 3-9.** Sample subset with imputed radar data and binary coding.......................... 64

**Table 3-10.** Sample radar data subset with no missing values, received after listwise deletion.......................................................... 65

**Table 3-11.** Sample radar data subset with no missing values (listwise deletion), after binary coding and normalisation. ......................................................... 66

**Table 3-12.** Classification performance (over the testing set) for continuous input coding and 12-10-N topologies with no data scaling, after normalisation and after standardisation. .......................................................... 68

**Table 3-13.** Classification performance (over the testing set) for binary input coding and 22-22-N topologies with no data scaling, after normalisation and after standardisation. .......................................................... 69

**Table 3-14.** Classification performance (over the testing set) for binary input coding and 22-22-11 topology with no data scaling, after normalisation and after standardisation. .......................................................... 71
TABLE 3-15. CLASSIFICATION PERFORMANCE (OVER THE TESTING SET) FOR CONTINUOUS INPUT CODING AND 12-10-N TOPOLOGIES WITH NO DATA SCALING, AFTER NORMALISATION AND AFTER STANDARDISATION. COMPARISON BETWEEN NN TRAINING WITH DATA RECEIVED AFTER LISTWISE DELETION AND AFTER MULTIPLE IMPUTATION. ................................................................................................................. 72

TABLE 3-16. CLASSIFICATION PERFORMANCE (OVER THE TESTING SET) FOR BINARY INPUT CODING AND 22-22-N TOPOLOGIES WITH NO DATA SCALING, AFTER NORMALISATION AND AFTER STANDARDISATION. COMPARISON BETWEEN NN TRAINING WITH DATA RECEIVED AFTER LISTWISE DELETION AND AFTER MULTIPLE IMPUTATION. ................................................................................................................. 74

TABLE 3-17. CLASSIFICATION PERFORMANCE (OVER THE TESTING SET) FOR BINARY INPUT CODING AND 22-22-11 TOPOLOGY WITH NO DATA SCALING, AFTER NORMALISATION AND AFTER STANDARDISATION. COMPARISON BETWEEN NN TRAINING WITH DATA RECEIVED AFTER LISTWISE DELETION AND AFTER MULTIPLE IMPUTATION. ................................................................................................................. 74
List of Figures

Figure 2-1. Texture classes’ distribution.................................................................................................................. 15

Figure 2-2. Samples of the acquired texture data - images of seven different types of wall cork tiles: Beach, Corkstone, Desert, Lisbon, Pebble, Precision and Speckled. ........ 15

Figure 2-3. Auto-associative neural network used to perform NLPCA. It has three hidden layers, including an internal bottleneck layer of a smaller dimension than either the input or output layers.................................................................................................................. 23

Figure 2-4. Self-organising map. Each output node is directly connected to all inputs. Each unit is given a weight vector that characterise the weights of those connections. ........................................................................................................................................... 29

Figure 2-5. Self-organising maps learning. The weights of the best matching unit and the neurons in its neighbourhood are adjusted towards the input vector. .................... 30

Figure 2-6. Within- and between-class variance, considering 3 of the 34 features of the original data (1st Co-occurrence matrix feature, 1st Laws’ mask feature and the Entropy feature). ................................................................................................................................. 33

Figure 2-7. Within- and between-class variance, considering the first 3 PCs, received after PCA with correlation matrix.................................................................................................................. 33

Figure 2-8. Within- and between-class variance, according to the first 3 KPCA components. (a) Scatter plot of the transformed data samples from the seven classes. (b) Texture classes means with 95% confidence intervals......................... 34

Figure 2-9. Within- and between-class variance, considering the first 3 NLPCA components. (a) Scatter plot of the transformed data samples from the seven classes. (b) Texture classes means with 95% confidence intervals......................... 35

Figure 2-10. Within- and between-class variance, considering the first 2 NLPCA components. (a) Scatter plot of the transformed data samples from the seven classes. (b) Texture classes means with 95% confidence intervals......................... 36

Figure 2-11. Example of texture images classified wrongly in the same SOM node: (a) Corkstone sample and (b) Lisbon sample. .................................................................................................................................................. 38
Figure 2-12. Sample self-organising map clustering: (a) after KPCA feature extraction and (b) after NLPCA feature extraction. ................................................................. 39

Figure 2-13. Sample SOM between-node distances map for NLPCA run with non-linear activation function in the bottleneck layer. Darker colours represent greater distances........................................................................................................ 39

Figure 2-14. Classification confusion matrices: (a) after KPCA feature extraction; (b) after NLPCA feature extraction. ................................................................. 40

Figure 2-15. Sample self-organising map clustering with colour blurring, reflecting the class probabilities for each neighbouring cell. (a) after KPCA feature extraction. (b) after NLPCA feature extraction. ................................................................. 42

Figure 3-1. A block diagram of a basic radar system. Radars operate by transmitting electromagnetic energy toward targets and processing the observed echoes. .... 47

Figure 3-2. Tangent sigmoid transfer function. ................................................................. 67

Figure 3-3. Investigated NN topologies for Case Study 1: 12 neurons in the input layer, 10 in the hidden and 1 (a) or 2 (b) neurons in the output layer.............................. 67

Figure 3-4. Classification results for 12-10-2 NN classifier with normalised input data and a validation stop after 118 epochs. The values in green specify the correctly classified samples for each class (10 - CIVIL, 01 - MILITARY). ........................................ 69

Figure 3-5. Classification results for 22-22-2 NN classifier with normalised input data and a validation stop after 114 epochs. The values in green specify the correctly classified samples for each class (10 – “CIVIL”, 01 – “MILITARY”). ......................... 70


Figure 3-7. Classification results for imputed data case for 12-10-2 NN classifier with normalised input data and a validation stop after 106 epochs. The values in green specify the correctly classified samples for each class (10 – “CIVIL”, 01 – “MILITARY”). ................................................................................................................................. 73

FIGURE 4-1. SULPHUR PREDICTION PERFORMANCE OF THE HYDROTREATING MODEL. IDEALLY, THE CONDITION POINTS SHOULD LIE ON THE DIAGONAL LINE (I.E. THE MEASURED VALUES TO COINCIDE WITH THE PREDICTED ONES). GAPS OF MORE THAN 50% OF THE MEASURED VALUES ARE OBSERVED. ................................................................. 85

FIGURE 4-2. NITROGEN PREDICTION PERFORMANCE OF THE HYDROTREATING MODEL. IDEALLY, THE CONDITION POINTS SHOULD LIE ON THE DIAGONAL LINE (I.E. THE MEASURED VALUES TO COINCIDE WITH THE PREDICTED ONES). GAPS OF UP TO 60PM FROM THE MEASURED VALUES ARE OBSERVED. ................................................................. 86

FIGURE 4-3. MAIN SCREEN OF THE GUI TOOL, DEVELOPED IN MATLAB® FOR ANALYSIS, OPTIMISATION, TESTING AND VALIDATION OF THE INVESTIGATED PREDICTION MODEL. .......... 88

FIGURE 4-4. PRODUCT SULPHUR OPTIMISATION RESULTS FOR A SAMPLE FEED. THE MEAN PREDICTION GAP IS ALMOST HALVED. .................................................................................. 91

FIGURE 4-5. PRODUCT SULPHUR OPTIMISATION RESULTS FOR ALL STUDIED FEEDS. THE MEAN PREDICTION GAP IS ALMOST HALVED. .................................................................................. 92

FIGURE 4-6. PRODUCT NITROGEN OPTIMISATION RESULTS FOR A SAMPLE FEED. THE MEAN PREDICTION GAP IS ALMOST HALVED. .................................................................................. 92

FIGURE 4-7. PRODUCT NITROGEN OPTIMISATION RESULTS FOR ALL STUDIED FEEDS. THE MEAN PREDICTION GAP IS ALMOST HALVED. .................................................................................. 93

FIGURE 5-1. RECTANGULAR PATCH APPLIED OVER A WING SHAPE AREA, SELECTED FOR OPTIMISATION. THE ARROWS SHOW THE DIRECTIONS OF THE LOCAL SURFACE NORMALS ............................... 102


FIGURE 5-3. A SCREENSHOT OF THE FIRST GUI TOOL, DEVELOPED IN MATLAB. FUNCTIONALITY FOR DERIVATION OF AN ARBITRARY NUMBER OF BASIS FUNCTIONS THAT ARE ORTHONORMAL WITH RESPECT TO A CHOSEN SURROGATE OBJECTIVE FUNCTION IS IMPLEMENTED. ................................. 111
FIGURE 5-4. BASIS FUNCTIONS IN CROSS-STREAM DIRECTION: a) even and b) odd functions..... 112
FIGURE 5-5. BASIS FUNCTIONS IN STREAMWISE DIRECTION: a) even and b) odd functions........ 112
FIGURE 5-6. BASIS FUNCTIONS EXPLORATION TOOL: a) GENERATED DISPLACEMENT AND b) STREAMWISE CURVATURE OF THE GENERATED DISPLACEMENT.................................................. 113
FIGURE 5-7. A SCREENSHOT OF THE SECOND GUI TOOL, DEVELOPED IN MATLAB. A MANUAL EXPLORATION OF THE EFFECTS OF VARYING A SINGLE OR A SET OF DESIGN VARIABLES OVER THE PATCH SURFACE IS SUPPORTED. BASIC CONSTRAINT CALCULATION IS IMPLEMENTED. .......... 114
FIGURE 5-8. WING SURFACE PERTURBATION: a) initial (red) and desired (green) shapes; b) fitted surface by 8 BASIS FUNCTIONS IN STREAM WISE AND 8 BASIS FUNCTIONS IN CROSS-STREAM DIRECTIONS. .................................................................................................................. 115
FIGURE 5-9. ZOOMED OSCILLATIONS OF THE GENERATED PATCH SURFACE. ALTHOUGH THE RESULTING SURFACE (RECEIVED AS A WEIGHTED SUM OF TENSOR PRODUCTS OF ORTHONORMAL BASIS FUNCTIONS MIGHT LOOK UNEVEN WHEN ZOOMED, THE OSCILLATIONS ARE WITHIN 0.5 MM AMPLITUDE DEVIATION OF THE DESIRED SURFACE. THE UNITS ON THE AXIS ARE METERS. ....... 115
FIGURE 5-10. CFD OBJECTIVE FUNCTION CONVERGENCE RATE, WHEN THE INVESTIGATED ORTHONORMALITY EXPLOITATION APPROACH IS NOT FOLLOWED. ................................................................. 117
FIGURE 5-11. CFD OBJECTIVE FUNCTION CONVERGENCE RATE, WHEN THE INVESTIGATED ORTHONORMALITY EXPLOITATION APPROACH IS APPLIED WITH BASIS FUNCTIONS, CALCULATED TO BE ORTHONORMAL WITH RESPECT TO THE STREAMWISE CURVATURE OF THE INVESTIGATED REGION................................................................. 117
FIGURE 5-12. CFD OBJECTIVE FUNCTION CONVERGENCE RATE, WHEN THE INVESTIGATED ORTHONORMALITY EXPLOITATION APPROACH IS APPLIED WITH BASIS FUNCTIONS CALCULATED TO BE ORTHONORMAL WITH RESPECT TO THE CHANGE OF THE STREAMWISE CURVATURE. ....... 118
## Glossary

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>AANN</td>
<td>Autoassociative Neural Networks</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>CFD</td>
<td>Computational Fluid Dynamics</td>
</tr>
<tr>
<td>ESM</td>
<td>Electronic Support Measures</td>
</tr>
<tr>
<td>GA</td>
<td>Genetic Algorithms</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>ICA</td>
<td>Independent Component Analysis</td>
</tr>
<tr>
<td>KPCA</td>
<td>Kernel Principal Component Analysis</td>
</tr>
<tr>
<td>LDA</td>
<td>Linear Discriminant Analysis</td>
</tr>
<tr>
<td>MI</td>
<td>Multiple Imputation</td>
</tr>
<tr>
<td>ML</td>
<td>Maximum Likelihood</td>
</tr>
<tr>
<td>MLP</td>
<td>Multilayer Perceptron</td>
</tr>
<tr>
<td>MSE</td>
<td>Mean Squared Error</td>
</tr>
<tr>
<td>NLPCA</td>
<td>Non-Linear Principal Component Analysis</td>
</tr>
<tr>
<td>NN</td>
<td>Neural Network</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>PDE</td>
<td>Partial Differential Equation</td>
</tr>
<tr>
<td>SOM</td>
<td>Self-Organising Map</td>
</tr>
<tr>
<td>SAR</td>
<td>Synthetic Aperture Radar</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial Basis Function</td>
</tr>
</tbody>
</table>
Chapter 1  Introduction

1.1  Motivation

In 1950, one of the pioneering computer scientists, A. M. Turing, suggested that intelligence is a matter of behaviour capacity and whether a system has a mind, or how intelligent it is, is determined by what it can or cannot do (Haugeland, 1997). He also proposed a pragmatic criterion of what a system can do that would be sufficient to show that it is intelligent, well known nowadays as the Turing test. Turing cast his test in terms of simulation or imitation: a non-human system will be deemed intelligent if it acts so like an ordinary person in certain respects that ordinary people cannot tell (from these actions) that it isn’t one (Haugeland, 1997).

A distinguishing feature of an intelligent system is that it has the ability to learn (Marsland, 2011). Therefore, the goal of building systems that can adapt and learn to recognise patterns from their experience has attracted researchers from many fields, including computer science, engineering, physics, and mathematics. This research has led to a wide variety of learning techniques and intelligent methods for solving a broad range of problems, ranging from computer vision to timely threat detection and avoidance, and from robotics to process automation and optimisation (Theodoridis & Koutroumbas, 2009).

The recent developments in the computational intelligence field have been driven by the enlarged availability of inexpensive computational resources, the
development of sophisticated machine learning algorithms, greater interest in
the area from both the research community and the industrial sector, as well as
the “data deluge” of an exponentially increasing quantity of data being
collected for processing (Bishop, 2013). However, there is still an abundance of
large-scale problems that continue to challenge scientists and different
industries. Some of their typical characteristics are: high-dimensional
parameter spaces, large amount of data, complexity and non-linearity
(Tsurkov, 2001; Biegler, Biros, & Ghattas, 2011).

In the applied science and technology, the ability to train models to recognise
patterns and to optimise those models to make accurate predictions is also a
topic of significant interest. Large number of machine learning techniques have
been developed, such as logistic regression, decision trees, support vector
machines, neural networks, Kalman filters and many others. Contributions to
this multi-disciplinary effort have come from the fields of statistics, signal
processing, artificial intelligence, optimisation, control theory, as well as from
the machine learning community itself (Bishop, 2006). Some of the most
common techniques in artificial intelligence can be found in the works of:
Bishop (1995); Bishop (2006); Theodoridis and Koutroumbas (2009); Umbaugh
(2010); Poole and Mackworth (2010); Abraham, Hassanien, Carvalho and
Snášel (2009); Hassanien and Abraham (2008) and Shih (2010).

Artificial intelligence (AI) systems provide a way to solve problems that are
complex and not always well defined. They are fault tolerant, as they can
handle noisy and incomplete data, and are able to learn from examples and
deal with non-linear problems. Once trained AI systems perform prediction
and generalisation at high speed (Kalogirou, 2003). AI systems include areas
such as artificial neural networks, genetic algorithms, fuzzy logic, expert
systems and a variety of hybrid systems, combining different techniques and
have been used in multiple applications in pattern recognition, prediction and
forecasting, optimisation, medicine, power systems, manufacturing, signal processing, control, etc.

The research presented in this thesis is motivated by the desire to investigate and apply novel pattern recognition and model optimisation techniques for addressing and providing solutions to large-scale real-world problems. In particular, a number of computational intelligence methods are proposed, implemented, tested, validated and evaluated in the search for convenient and reliable solutions to several challenging industrial tasks (organised in self-contained chapters due to their diversity). A discussion on the motivation for investigating those problems is provided at the beginning of each of the following chapters and the original outcomes of this research are summarised in section 1.2.

1.2 Original Contributions to Knowledge

Four computational intelligence systems for solving large-scale real-world problems in the fields of pattern recognition and optimisation are investigated, implemented, tested, validated and evaluated in this dissertation. In their essence, the original contributions to knowledge can be summarised as follows:

- The functionality of an intelligent visual inspection system for classification of textural images is extended with unsupervised learning capabilities. Several dimensionality reduction techniques are considered and a novel classification approach, incorporating autoencoders-based non-linear principal component analysis (NLPCA) method and self-organising maps, is proposed, developed and applied for solving the investigated task;
• Several methods for dealing with the large amount of missing values in intercepted radar signal data are proposed for solving a classification problem for timely and reliable identification of the signal emitters. Approaches incorporating listwise deletion, multiple imputation and neural networks are investigated, implemented, tested and validated on a number of case studies and the results are evaluated and critically compared;

• Global optimisation, incorporating genetic algorithm techniques, is performed over a number of the setup parameters of a kinetic prediction model used in British Petroleum refineries. The model simulates the effects different crude oils have on the lifecycle of the loaded catalyst during a major purifying process (known as hydrotreating). Several MATLAB functions and a graphical user interface (GUI) tool are designed and implemented. The simulation, testing and validation results show significant improvement in the accuracy and efficiency of the optimised model (in comparison to the one, initially used by BP);

• A novel approach for parametric patch perturbation for automated optimisation of aerodynamic shapes is presented, implemented, tested and evaluated. Physics-grounded heuristics are explored to derive a set of basis functions (for spanning the design space) that can significantly improve the convergence rate of a very slow computational fluid dynamics (CFD) based optimisation process. Several MATLAB functions and two graphical user interface (GUI) tools are implemented to support the basis functions calculation, displacement patch exploration, optimisation, evaluation and visualisation.
The above-listed contributions have been disseminated by:

- Presenting the research outcomes to five international conferences and congresses;
- Preparing four journal papers, two of which have been published and two are currently under review;

1.3 Thesis Outline

This thesis is organised in six chapters. The first chapter discusses the motivation for conducting this research work (section 1.1), including a section (1.2) on the original contributions made and on the organisation of this thesis (section 1.3). Due to the large scope and diversity of problems addressed in this research, each of the next four chapters is prepared and presented as a self-contained study on a particular problem (given in detail in each chapter) from the pattern recognition and model optimisation areas.

*Chapter 2* investigates an intelligent visual inspection system for pattern recognition and classification of texture images. It starts by a literature review on the studied problem (section 2.1) and provides a brief discussion (section 2.2) on the data acquisition, feature extraction and statistical pre-processing stages of the proposed approach. The chapter continues by giving detailed information on the used linear and non-linear dimensionality reduction methods (section 2.3), followed by an overview of the proposed unsupervised learning approach used in the classification stage (section 2.4). The obtained
results are summarised and evaluated in section 2.5 and conclusions and ideas for further work are given in section 2.6.

*Chapter 3* studies a classification problem for timely and reliable identification of radar signal emitters by implementing and following a neural network based approach. The chapter starts by a review of related literature on the investigated topic (section 3.1), followed by formulation of the problem and an analysis of the available data (section 3.2). Different statistical transformation techniques are discussed and a multiple imputation method for dealing with missing data is introduced in section 3.3. Several neural network topologies, training parameters, input and output coding, and data transformation techniques for facilitating the learning process are tested and evaluated on a set of case studies in section 3.4. Finally, section 3.5 summarises the results and provides ideas for further extension of this research.

*Chapter 4* investigates a model optimisation approach for improving the accuracy and efficiency of an industrial prediction model used in petroleum refineries in order to meet the increased demand for cleaner fuels and petroleum products. The chapter starts by providing relevant background knowledge and a review of related literature on the investigated problem (section 4.1), followed by a section on the current state of the studied simulation model (section 4.2) and a discussion on the adopted data acquisition and model gap identification steps (section 4.3). The undertaken optimisation approach is described in section 4.4, alongside with details on the development of a MATLAB-based GUI tool to assist the process. The final two sections present the results from the optimisation (4.5) and conclude the chapter, providing ideas for further extension of this work (4.6).

*Chapter 5* introduces a novel efficient method for automated optimisation of aerodynamic shapes based on displacement patch generation. The chapter starts by literature review (section 5.1) of the limited research available on the investigated topic. Next, the key ideas behind the implemented solution
(section 5.2) are presented, followed by a section (5.3) discussing in detail the suggested approach. The derivation of the used basis functions and the implemented software solutions are presented in section 5.4 and section 5.5, respectively. The initial results from running large simulations with real aircraft design surfaces are given in section 5.6. Section 5.7 summarises the chapter and gives ideas for further extension of this work.

Finally, chapter 6 concludes the thesis by summarising the contributions made in this research work (section 6.1), outlines ideas for further investigation (section 6.2) and gives a list of peer-reviewed publications that disseminate the key research outcomes.
Chapter 2  Intelligent Pattern Recognition and Classification for Texture Images

This chapter investigates an intelligent visual inspection system for pattern recognition and classification of texture images. The task is to extend an existing system, developed to use expert knowledge in the training process (supervised learning), with capabilities for unsupervised learning (where no \textit{a priori} knowledge is used) to automate the classification of different designs of cork tiles.

For each sample, a set of textural features is derived using spatial (\textit{Grey-Level Co-occurrence matrices}), filtering (\textit{Laws’ masks}) and statistical (\textit{Entropy}) techniques. In order to reduce the dimensionality of the feature dataset and consequently the computational load of the classification process, an approach incorporating Non-Linear Principal Component Analysis (NLPCA) is proposed and applied to the extracted feature vectors. Self-Organising Map neural networks are employed for solving the classification problem. A number of experiments and simulations are conducted to evaluate the proposed methods and the results are critically analysed and compared with the ones reported by authors working on similar topics.

The rest of the chapter is organised as follows: in the next section (2.1) a review of the background knowledge used in this chapter, as well as the state of the
art for the investigated problem are given. Section 2.2 presents a brief discussion on the data acquisition, feature extraction and statistical pre-processing stages of the proposed approach. The chapter continues by giving detailed information on the used linear and non-linear dimensionality reduction methods (section 2.3), followed by introduction of the proposed unsupervised learning approach used in the classification stage (section 2.4). The obtained results are summarised and evaluated in section 2.5 and conclusions and ideas for further work are given in section 2.6.

2.1 Background Knowledge

Texture description and recognition is becoming more and more integrated in industrial applications for quality and process control (Costa & Pereira, 2006; Egmont-Petersen, de Ridder, & Handels, 2002; González, Bianconi, Álvarez, & Saetta, 2013; Ortiz-Jaramillo, Orjuela-Vargas, Van-Langenhove, Castellanos-Domínguez, & Philips, 2014; X. Wang, Georganas, & Petriu, 2011; X. Xie, 2008). In particular, the use of pattern recognition can have a strong impact on automated visual inspection process, where decisions about the quality of the products have to be made constantly and in real time (Davies, 2012). In the literature, a number of different paradigms, methods and algorithms have been developed (González et al., 2013; Martens, Poppe, Lambert, & Van de Walle, 2008; Paniagua, Vega-Rodríguez, Gomez-Pulido, & Sanchez-Perez, 2010; Shih, 2010; J. Xie, Zhang, You, & Shiu, 2015). In (Georgieva & Jordanov, 2009), for example, a texture image classification problem, using supervised neural networks (NN) has been investigated, where the textural features of each image have been derived by means of two classical feature extraction methods: filtering approach (Laws’ masks) and statistical approach (grey-level co-occurrence matrices). The effectiveness of these techniques has also been tested and proven on various problems (Davies, 2012; Ortiz-Jaramillo et al., 2014; X.
Xie, 2008). However, the feature vectors obtained by these two methods are usually too large to be used directly as input to a classifier, leading to the “curse of dimensionality” and other problems related to the generalisation abilities of the trained systems, especially when insufficient training samples are available. Moreover, a proper reduction of the number of variables representing the data helps to dispose of redundant information and has a positive impact on the computational load of the classification process. Many dimensionality reduction techniques are based on the use of principal component analysis (PCA) (Astel, Tsakovski, Barbieri, & Simeonov, 2007; Chamundeeswari, Singh, & Singh, 2009; Georgieva & Jordanov, 2009; Jimenez-Rodriguez, Arzuaga-Cruz, & Vélez-Reyes, 2007). However, while much has been learned using PCA and related techniques, the fact that they are linear methods implies a potential oversimplification of the datasets being analysed and in this way losing information and statistical power.

A possible solution for this problem can be achieved by using Nonlinear Principal Component Analysis (NLPCA). The NLPCA (Scholz, 2012) is a nonlinear generalisation of the PCA, but while the PCA is restricted to linear components, the NLPCA generalises the principal components from straight lines to curves and hence describes the inherent structure of the data by curved subspaces. NLPCA can be performed by using a multi-layer perceptron (MLP) with an auto-associative topology, also known as auto-encoder, replicator network, bottleneck, or sand-glass type network. Compared to the linear PCA, the NLPCA is able to represent the data structure with fewer components (G. Licciardi, Avezzano, Del Frate, Schiavon, & Chanussot, 2014).

**State of the Art**

Image analysis techniques have already been applied for automated visual inspection, product quality control and materials characterisation of variety of products in industry (González-Hernández, González-Adrados, de Ceca, & Sánchez-González, 2014; González et al., 2013; A. Kumar, 2008; Ortiz-Jaramillo
et al., 2014; Ruz, Estévez, & Ramirez, 2009; X. Wang et al., 2011), and in particular for cork products in (Costa & Pereira, 2006; Georgieva & Jordanov, 2009; Oliveira, Knapić, & Pereira, 2013; Paniagua et al., 2010; Radeva, Bressan, Tovar, & Vitria, 2002; Vitria, Bressan, & Radeva, 2007), where the investigated image-based inspection systems have shown high recognition rates. Such systems usually comprise of a line-scan camera and a computer, embedded in an industrial sorting machine, which is capable of acquiring and real-time processing of product surface images.

Radeva et al. (2002) considered intelligent visual inspection of five types of cork stoppers. Applying PCA, LDA, nearest neighbour classifier and independent component analysis (ICA) in combination with Bayesian classifier, they achieved higher success rate in all cases and especially for the ICA set. In a subsequent work (Vitria et al., 2007) they implemented and tested several parametric and non-parametric methods for cork stopper quality classification and again used PCA and LDA for reducing the feature spaces (they also expressed concerns that PCA and LDA may hide important information for the following classification process). In their tests, the non-parametric group of techniques showed inferior performance compared to the parametric ones (reporting the best result for a Naive Bayes classifier).

Costa and Pereira (2006) investigated a system for intelligent inspection of seven types of cork stoppers and reported overall low success rate due to possibly inadequate feature selection and lack of other than LDA classifiers, which apparently is not enough for problems with similar complexity.

Georgieva et al. (2009b) reported up to 85% classification accuracy for some of the cases, after applying PCA, linear discriminant analysis (LDA) and supervised neural network learning for classification of cork tiles images.

Oliveira et al. (2013) used stepwise discriminant analysis to build predictive classification models to characterise the surface of cork stoppers and cluster them into three quality classes and analyse the contribution of each porosity feature to the class classification.

Dimensionality reduction techniques are used to lower the complexity of an investigated problem and to increase the computational efficiency of subsequent pattern recognition and classification processes. In general, dimensionality reduction methods can be divided into two groups: feature-selection algorithms (which suitably select a sub-optimal subset of the original set of features while discarding the remaining ones), and feature extraction by data transformation (which project the original feature space onto a lower dimensionality subspace that preserves most of the original information) (Serpico, D’Inca, Melgani, & Moser, 2003). The feature selection is more direct and simpler approach in comparison to the feature extraction, and the resulting reduced set of features is easier to interpret. However, extraction methods can be expected to be more effective in representing the information content in lower dimensionality domain (Serpico et al., 2003).

Feature-selection techniques can be generally considered a combination of both a search algorithm and an objective function (Jimenez-Rodriguez et al., 2007; Somol & Pudil, 2002). The solution to the feature-selection problem is provided by the search algorithm, which generates a subset of features and compares them based on the objective function. From a computational viewpoint, an exhaustive search for the optimal solution becomes intractable even for moderate values of features. In addition, computational power is not enough to make it feasible for problems with hundreds of features. Despite these apparent difficulties, several other feature selection approaches have been
developed: feature similarity measures (Mitra, Murthy, & Pal, 2002), graph searching algorithms (Plaza, Martinez, Plaza, & Perez, 2005), neural networks (Marpu, Gamba, & Niemeyer, 2009), support vector machines (Melgani & Bruzzone, 2004), and others.

Feature-extraction techniques aim at reducing the data dimensionality by mapping the feature space onto a new lower-dimensional space. Both supervised and unsupervised methods have been developed.

Supervised feature extraction techniques directly take into account the training information available for the solution of a given supervised classification problem. Three main approaches based on discriminant analysis, decision boundary analysis, and correlated feature grouping, have been proposed in the literature (Bressan & Vitria, 2003; Jia & Richards, 1999; Warner, Steinmaus, & Foote, 1999). The first one is based on the maximisation of a functional (e.g., the Rayleigh) coefficient, expressed as the ratio of a between-class scatter matrix to an average within-class scatter matrix (Bressan & Vitria, 2003). This technique has some drawbacks, such as the possibility of extracting at most \((C-1)\) features, where \(C\) is the number of classes. The second approach employs information about the decision hyper-surfaces associated with a given parametric Bayesian classifier to define an intrinsic dimensionality useful for the classification problem and the corresponding optimal linear mapping. The third strategy consists of grouping the original features into subsets of highly correlated features to transform the features separately in each subset (Jia & Richards, 1999; Warner et al., 1999).

Unsupervised feature-extraction methods do not require any prior knowledge of the training data, even though they are not directly aimed at optimising the accuracy in a given task (Jimenez-Rodriguez et al., 2007). The class includes methods, such as PCA, ICA, and Maximum Noise Fraction (MNF), where an operator calculates a set of transformed features according to a signal-to-noise ratio optimisation criterion. In these techniques, the dimensionality reduction
is obtained discarding the less relevant components received after the projection into the feature space. However, since the projection is performed in a linear way, it is not always possible to detect all the relevant structures in a given data. Thus, it is desirable to use nonlinear transformations in order to extract more information. Many methods have been proposed to extract components in a nonlinear manner, e.g., Locally linear embedding (LLE) (Saul & Roweis, 2003) and Isomap (Tenenbaum, De Silva, & Langford, 2000) visualise high dimensional data by projecting (embedding) them into a two or three-dimensional space. Principal curves and self-organising maps (SOM) (Kohonen, 2001) describe data by nonlinear curves and nonlinear planes up to two dimensions. The main limitation of these methods is related to obtaining low number of features, that may be not sufficient to describe the inherent information of the data.

Kernel PCA (KPCA) and Nonlinear PCA (NLPCA) present a higher degree of freedom in terms of feature dimensionality (Schölkopf, Smola, & Müller, 1998). KPCA first performs a nonlinear transformation of the data into a higher dimensional space, and then uses PCA linear transformation in order to reduce the effective dimension of the data. Nonlinear PCA, on the other hand, performs the projection into a reduced feature space by means of autoassociative neural networks (AANN). Both methods could be considered as a nonlinear generalisation of the standard PCA and tend to produce similar results. However, in (G. Licciardi, Marpu, Chanussot, & Benediktsson, 2012) a comparison of PCA, KPCA and NLPCA, used for dimensionality reduction, demonstrates a better performance of the nonlinear dimensionality reduction techniques in terms of classification accuracy of the reduced dataset.
2.2 Data Acquisition and Preparation

The texture image data set used for this investigation is acquired via an intelligent visual recognition system described in more detail in (Georgieva & Jordanov, 2009). The system uses a charge-coupled device (CCD) camera, lightning devices, scaffolding and a computer. The dataset consists of 335 images of size 230x340 pixels of industrial cork tile samples, predefined by experts in seven types based on their texture - Beach, Corkstone, Desert, Lisbon, Pebble, Precision and Speckled. The distribution of the texture classes (non-uniform) is given in Figure 2-1 and samples of each type are presented in Figure 2-2.

![Figure 2-1. Texture classes' distribution.](image)

![Figure 2-2. Samples of the acquired texture data - images of seven different types of wall cork tiles: Beach, Corkstone, Desert, Lisbon, Pebble, Precision and Speckled.](image)
Since the texture of the samples is of prime interest, the images are converted to greyscale. In addition, in order to reduce the illumination effects on the analysed images (e.g., due to a glare, reflections, etc.), a normalisation technique is applied. In this process, a small window (15x15 pixels) is moved within each image and the local average is subtracted from the pixels’ values, in order to get images with average intensity of each neighbourhood about a zero (Davies, 2012).

Afterwards, a set of features is extracted using spatial, filtering and statistical techniques (Section 2.2.1), which are then rescaled (Section 2.2.2), in order to facilitate the subsequent analysis and learning stages (Section 2.4).

### 2.2.1 Features Extraction

The feature extraction phase aims to identify characteristics and properties that make the sample classes distinct from each other (Scholz, 2012). At this stage of the process, features that represent some valuable information about the texture of the images are obtained. Three different techniques are used for extracting a total of 34 features.

**Co-occurrence Matrices**

*Co-occurrence matrices* is a commonly applied statistical approach for texture features extraction that takes into account relative distances and orientation of pixels with co-occurring values (Davies, 2012; Ortiz-Jaramillo et al., 2014; Salah, Trinder, & Shaker, 2009).

The MATLAB® *Image Processing Toolbox* is used for the computation of the co-occurrence matrices of the normalised images. As usually proposed, four relative orientations are used – horizontal (0°), right diagonal (45°), vertical (90°), and left diagonal (135°). In this way, the *energy, homogeneity, correlation,*
and contrast characteristics in each direction are computed to obtain the rotation invariant features (Davies, 2012).

In addition, two spatial relationships are considered – the direct neighbours and the pixels with difference of five. As a result, a total of eight co-occurrence matrices are obtained – four for the direct neighbours and another four for the pixels with difference of five.

**Laws’ Masks**

The Laws’ masks are used as a filter technique that is applied to identify points of high energy in an image (Davies, 2012; Ortiz-Jaramillo et al., 2014). They are derived from three basic 1x3 masks:

\[
L_3 \text{ (Local averaging) } = [1 \ 2 \ 1],
\]

\[
E_3 \text{ (Edge detection) } = [-1 \ 0 \ 1],
\]

\[
S_3 \text{ (Spot detection) } = [-1 \ 2 \ -1],
\]

which represent one-dimensional operations of centre-weighted local averaging, symmetric first differencing (edge detection), and second differencing (spot detection) (Davies, 2012). After applying discrete convolution to those three simple vectors, five distinct masks (one dimensional vectors of five pixels length) for picking up the average grey level, edges, ripples, spots and waves are received:

\[
L_5 \text{ (Level detection) } = [1 \ 4 \ 6 \ 4 \ 1] = L_3*L_3,
\]
\[
E_5 \text{ (Edge detection) } = [-1 \ -2 \ 0 \ 2 \ 1] = L_3*S_3,
\]
\[
S_5 \text{ (Spot detection) } = [-1 \ 0 \ 2 \ 0 \ -1] = S_3*S_3,
\]
\[
R_5 \text{ (Ripple detection) } = [1 \ -4 \ 6 \ -4 \ 1] = L_3*E_3,
\]
\[
W_5 \text{ (Wave detection) } = [-1 \ 2 \ 0 \ -2 \ 1] = -E_3*S_3,
\]

where ‘*’ denotes the convolution operator.
These vectors are multiplied to each other and this way 25 different 5x5 masks are produced. The masks are then applied to the normalised set of samples and the obtained filtered images are converted to texture energy maps. In order to deduce the local magnitudes of the quantities of interest (edges, spots, etc.) a smoothing window of size 15x15 (Davies, 2012) is applied to each filtered image $F_k$ for the $k$-th mask resulting in new energy images, each pixel of which is given with:

$$E_k(r, c) = \sum_{i=r-p}^{i+p} \sum_{j=c-p}^{c+p} |F_k(i, j)|, \quad k = 1, ..., 25, \quad (2.1)$$

where $(r, c)$ denotes the scan position, $F_k(i, j)$ is the local magnitude of a microfeature $k$ in a $(2p+1) \times (2p+1)$ window. After obtaining energy maps for each image, a power metric, representing the sum of the squared absolute values for each pixel in the map is used, to finally obtain 25 different values for each texture sample.

**Entropy**

*Entropy* is a statistical measure of randomness that can be used to characterise the texture of an image (Davies, 2012; Kuo & Kao, 2007). It takes low values for smooth images and vice versa.

The entropy for each image sample is calculated using a MATLAB’s build-in function, as given below:

$$E = -\sum_{i=1}^{G} d(i) \log_2 d(i), \quad (2.2)$$

where $G$ is the number of grey levels in the image histogram, ranging between 0 and 255 for a typical 8-bit image, and $d(i)$ is the normalised occurrence frequency of each grey level.
2.2.2 Scaling

The extracted 34 textural features (8 co-occurrence matrices, 25 energy maps, and the entropy) present different scales and thus a scaling pre-process is necessary. This is particularly useful when using neural networks, where the data range of each input should be projected to the linear part of the activation function (Bishop, 1995).

Normalisation and standardisation are widely used simple statistical approaches for rescaling data, but they both have their drawbacks. In case of the data having outliers, the normalisation will compress the ‘normal’ data in a smaller interval. On the other hand, standardisation assumes Gaussian distribution of the data, which may not be the case.

Since most of the real-world datasets contain outliers and have non-Gaussian distribution, the use of standardisation or normalisation separately may be counterproductive. When investigating neural networks training and learning, a common approach is to combine both techniques, by data being firstly normalised and then standardised. However, the presence of outliers may result in a non-Gaussian distribution leading to non-efficient standardisation. To overcome this problem, an alternative standardisation that uses the median instead of the mean value is used:

\[ x' = \frac{x_{\text{norm}} - m_{\text{norm}}}{\sigma_{\text{norm}}} \]  \hspace{1cm} (2.3)

where \( x_{\text{norm}} \) is the normalised in (-1, 1) interval data, \( m_{\text{norm}} \) is the median, and \( \sigma_{\text{norm}} \) is the standard deviation.
2.3 Dimensionality Reduction

Before applying any statistical analysis and feature reduction, a random subset of 30% of the available data is excluded for subsequent testing. This subset will be referred to as the testing set from now on and the remaining 70% of the available data will be the training set.

During the feature extraction stage, a total of 34 features are obtained for each texture image (8 by the co-occurrence method, 25 by Law’s masks and 1 entropy feature).

In order to reduce the dimensionality of the classification problem (i.e., the number of inputs to the classifier), to dispose of the redundant information (i.e., the information contained in some highly correlated features) and to improve the class separability, an approach based on the nonlinear generalisation of the standard principal component analysis (PCA) is proposed. In the literature, there are many other methods which visualise data and extract components in a nonlinear manner (Gorban, 2007). Although some supervised dimensionality reduction techniques (such as Linear Discriminant Analysis (LDA)) are used for comparison purposes in some of the experiments, since the one of the aims of this chapter is to investigate the influence of nonlinear dimensionality reduction on the performance of the intelligent system, the focus will be maintained on two main nonlinear approaches: nonlinear PCA and Kernel PCA.

2.3.1 Principal Component Analysis (PCA)

*Principal component analysis* (PCA) is an eigenvalue-based multivariate technique that transforms a number of possibly correlated features into a number of uncorrelated features, called principal components (PCs) (Egmont-
Petersen et al., 2002; Paniagua et al., 2010). The number of the derived PCs is less than or equal to the number of the original features. It is an unsupervised technique and as such does not use any labelled information on the data.

The first PC accounts for as much of the variability (information) in the data, as possible, and each succeeding PC accounts for as much of the remaining variability as possible. Depending on the areas of application, PCA is also referred to as Hotelling transform, Karhunen-Loeve transform (KLT), or proper orthogonal decomposition (POD) (Paniagua et al., 2010).

### 2.3.2 Nonlinear Principal Component Analysis (NLPCA)

Nonlinear Principal Component Analysis (NLPCA) is commonly performed by an Autoassociative Neural Network (AANN) or autoencoder. NLPCA is initially introduced by Kramer (Kramer, 1991) as an effective instrument for data reduction and decorrelation of different types of data. In (Del Frate, Iapaolo, Casadio, Godin-Beekmann, & Petitdidier, 2005) the dimensionality of huge atmospheric datasets is reduced with an AANN for the retrieval of atmospheric variables, while in (G. Licciardi et al., 2014) a similar approach is used to decompose different scattering mechanisms in polarimetric synthetic aperture radar (SAR) images. In (G. Licciardi et al., 2012; G. A. Licciardi et al., 2012) NLPCA is used to reduce the spectral dimensionality of different hyperspectral images, while in (G. A. Licciardi & Del Frate, 2011) the resulting nonlinear principal components extracted from an AANN are used as abundance maps in an unsupervised nonlinear spectral unmixing approach.

An autoencoder is a conventional feedforward neural network having sigmoidal activation functions in each node and the training of the AANN is performed by standard back-propagation or similar algorithms (Bishop, 1995).
In particular, in this work the chosen training algorithms is the Scaled Conjugated Gradient (SCG). The standard autoencoder presents a symmetrical three layer topology, where input and output layers have the same number of nodes, and a hidden layer, usually referred as bottleneck, of smaller dimension than either input/output layers.

The AANN is trained to perform identity mapping (the input $X$ has to be equal to the output $Y$), in order to minimise the sum-of-squares error (MSE) of the form:

$$MSE = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{d} [y_d(x^n) - x_k^n]^2,$$

where $X = \{x_1, x_2, \ldots, x_k\}$ and $Y = \{y_1, y_2, \ldots, y_k\}$ are the input and output vectors, respectively, $d$ is the number of output neurons and $N$ is the number of training samples.

This means that if the training phase finds an acceptable solution (e.g., the MSE reaches a value that is under a certain threshold), the autoencoder is able to reconstruct in the output a good representation of the input and since there are fewer nodes in the bottleneck layer than in the input/output, the bottleneck nodes represent or encode the information obtained from the inputs for the subsequent layers to reconstruct the output. In other words, data compression caused by the network bottleneck forces hidden units to represent significant features in the data, removing redundancies.

Different from standard autoencoders, the topology of a nonlinear AANN by default uses three hidden layers, including the internal bottleneck layer of smaller dimension than either input or output layers (Figure 2-3). In order to understand why three hidden layers are necessary to obtain a nonlinear representation of the data it is useful to consider the nonlinear AANN as a combination of two successive neural networks or functional mappings. The first sub-network represents the encoding or extraction function:
\[ T = F(X) \] (2.5)

that projects the original \( k \)-dimensional data onto a lower dimensional subspace defined by the activations of the units in the central hidden layer (bottleneck). Similarly, the second sub-network defines an arbitrary functional mapping:

\[ Y = G(T) \] (2.6)

that projects from the low-dimensional feature space back into the original \( k \)-dimensional space. Each of these two subnetworks has the ability to fit arbitrarily nonlinear functions thanks to the presence of a hidden layer with nonlinear nodes between input and bottleneck layer. An AANN without the coding and decoding layers is able to fit only linear functions and corresponds exactly to the linear PCA. This explains why three hidden layers are necessary in order to obtain an optimal nonlinear feature extraction.

![Diagram of an auto-associative neural network](image)

**Figure 2-3.** Auto-associative neural network used to perform NLPCA. It has three hidden layers, including an internal bottleneck layer of a smaller dimension than either the input or output layers.
Training an AANN is not an easy task because, apart from the number of features, there is no definitive method for selecting the dimensions of the mapping and demapping layers. Since the number of nodes in the coding/decoding layers is related to the complexity of the functions describing the original data, the representation in the feature space can be strongly constrained if too few nodes are used in the two mapping layers. On the other hand, if too many nodes are used, the AANN may become prone to overfitting.

In order to minimise the reconstruction error, the best NN topology can be retrieved by using grid-search algorithms that vary recursively the number of nodes in the hidden layers and evaluate the respective error. Then, the topology presenting the lowest error is selected.

A way to determine the correct dimension of the bottleneck layer is to analyse the correlation between the different NLPCs. In particular, it has been noted that if the number of nodes in the bottleneck layer is lower than the intrinsic dimensionality of the data, the training phase will not be able to reach appreciable MSE values. On the contrary, if the dimension of the bottleneck layer is bigger than the intrinsic dimensionality of the data, two or more NLPCs tend to be extremely correlated. This characteristic can be exploited through the use of a grid search algorithm, where once the number of nodes in the coding/decoding layers is set to an arbitrary value, it varies recursively the number of nodes in the bottleneck layer and evaluates the correlation between the components. The highest number of nodes in the bottleneck layer not presenting correlation between components is then selected.

Once the correct dimensionality of the feature space is detected, a further grid search algorithm varying the dimension $M$ of the coding/decoding layers is selected. Thus the AANN topology presenting the lowest MSE is defined and the NLPCs can be extracted.
2.3.3 Hierarchical NLPCA

In general, when using a neural network approach for the production of a linear or nonlinear subspace decomposition it is important that the derived features are unique. The only requirement to obtain such uniqueness is that the subspace retains as much information as possible, expressed in terms of MSE. This may lead to more than one solution. This problem can be solved by enforcing a hierarchical order of the principal components (H-NLPCA) (H. Liu & Yu, 2005). There are two ways to introduce hierarchical constraints in the feature space. The first one is to force the \( i \)-th feature to account of the \( i \)-th highest variance projection. However, this approach can be extremely difficult due to bounding conditions. Another approach is to search for the lowest MSE in the original data space using only the first \( i \) features. This can be obtained by training the AANN considering not only the minimisation of the general MSE, but also the MSE obtained using only a subset of features. This can be done by minimizing the hierarchical error, defined as:

\[
E_{H} = E + \sum_{i=1}^{N-1} \alpha_i E_{1,i},
\]

where \( E \) is the general MSE evaluated on \( N \) features, \( E_{1,i} \) are the MSE errors for the features subsets and \( \alpha_i \) are weighting parameters. However, the selection of the optimal \( \alpha_i \) may lead to an unacceptable increase of computational load. Experimental studies demonstrate that good solutions can be achieved also setting \( \alpha_i=1 \) (H. Liu & Yu, 2005). Since the different errors should be calculated separately, \( N \) AANN subnetworks have to be evaluated separately with a varying number of nodes in the bottleneck (Figure 2-3). Thus, the gradient \( \nabla E_{H} \) is the sum of the single gradients obtained from each AANN:

\[
\nabla E_{H} = \nabla E + \sum_{i=1}^{N-1} \nabla E_{1,i},
\]
The above mentioned hierarchical condition implies that instead of searching for a $N$-dimensional subspace that minimise the general MSE, the minimisation of the general MSE is constrained to the minimisation of the MSE evaluated on a $(N-1)$-dimensional subspace. This is successively extended, such that all $1,...,N$ dimensional subspaces are of minimal MSE. In this way, each subspace represents the data with regard to its dimensionalities (H. Liu & Yu, 2005).

In order to differentiate the two above-mentioned NLPCA methods, we shall refer to “hierarchical” NLPCA when the training is constrained to the minimisation of both general and subspace MSE. On the other hand, when the training phase is based only on the minimisation of the general MSE it will be referred as “symmetrical” NLPCA.

### 2.3.4 Kernel PCA

Even though the term nonlinear PCA (NLPCA) is often referred to the auto-associative neural network approach, Kernel Principal Component Analysis (KPCA) offers an alternative way to perform the nonlinear generalisation of PCA. KPCA is a method that uses integral operator kernel functions to perform a nonlinear form of PCA (Schölkopf et al., 1998).

In linear PCA, it is possible to define a linear mapping function that projects input data $x_i$, having dimension $d$, into a feature space $N$, where $d>N$:

$$
\Phi(x_i) \text{ where } \Phi : \mathbb{R}^d \rightarrow \mathbb{R}^N.
$$

Since the PCA is not able to deal with nonlinearities of the original data, part of the relevant information may not be retained by the first $k$ components, and consequently can be lost as consequence of the dimensionality reduction. However, the basic idea of KPCA, in order to capture higher-order statistics, is
to map the data with a function that can be linear or nonlinear, onto another space $H$:

$$\Phi(x_i) \text{ where } \Phi : \mathbb{R}^d \rightarrow H.$$  \hspace{1cm} (2.10)

The space $H$ must have the structure of a Reproducing Kernel Hilbert Space (RKHS), not necessarily of finite dimension. At this point the projected data into the feature space $H$ is no more affected by nonlinearities and PCA can be performed as in the input space to reduce the dimensionality without losing relevant information. However, since the space $H$ may have a dimension $m \gg d$, performing the PCA may be extremely difficult from a computational point of view. To avoid this problem, it is possible to use the kernel trick that permits to perform the PCA directly in the input space. The kernel PCA solves the following eigenvalues problem:

$$\lambda \alpha = K \alpha, \text{ where } \|\alpha\|_2 = \frac{1}{\lambda} \hspace{1cm} (2.11)$$

and $K$ is the kernel matrix defined as:

$$K = \begin{pmatrix}
    k(x^1, x^1) & \cdots & k(x^1, x^d) \\
    k(x^2, x^1) & \cdots & k(x^2, x^d) \\
    \vdots & \ddots & \vdots \\
    k(x^d, x^1) & \cdots & k(x^d, x^d)
\end{pmatrix}. \hspace{1cm} (2.12)$$

The function $k(\cdot)$ is the core of the KPCA and is usually called kernel. It is a positive semi-definite function on the input space that introduces nonlinearity into the processing.
2.3.5 Linear Discriminant Analysis

Linear discriminant analysis (LDA) is an eigenvalues-based transformation technique that aims to find a linear combination of features that characterise or separate two or more classes (Dillon & Goldstein, 1984; Umbaugh, 2010). LDA is not used in this work as a classification technique, but as a data pre-processing transform, before applying the classification technique, as recommended in (Bishop, 1995). The number of the newly generated features is always one less than the number of the classes. This approach is considered for comparison purposes and it is not of main focus of this chapter.

2.4 Classification

For the classification of the texture samples data, self-organising maps (SOMs) are employed. They are also known as Kohonen maps or self-organising feature maps and represent a type of an artificial neural network (NN), proposed by Teuvo Kohonen in the late 1980s (Kohonen, 2001). SOMs are trained using unsupervised learning to produce a low-dimensional (typically two- or three-dimensional), discretised representation of the input space of the training samples, called a map. A specific characteristic of SOMs (compared to other NNs) is that they use a neighborhood function to preserve the topological properties of the input space (Kohonen, 2001). As a result, a cluster map is received, in which the presented distances and relationships are proportional to the distances and the relationships between the input patterns.

A typical SOM consists of a 1- (string structure), 2- (array structure) or 3-dimensional (parallelepiped structure) grid of simple units (neurons), each node of which is directly connected to all inputs (Figure 2-4). Also, each unit is given a weight vector (with the same size as the input data vectors), the
elements of which characterise the weights of those connections. These vectors are randomly initialised, i.e. the units are randomly distributed in the input space (Davies, 2012).

![Self-organising map](http://www.lohninger.com/kohonen.html)

Figure 2-4. Self-organising map. Each output node is directly connected to all inputs. Each unit is given a weight vector that characterise the weights of those connections.  

Similar to most neural networks, SOMs operate in two modes: training and testing. During the training stage, all weight vectors are computed for each training sample. The neuron whose weight vector is most similar to the input vector is selected as best matching unit (BMU). Next, the weights of the BMU and the neurons in its neighbourhood are adjusted towards the input vector. The amount of change introduced decreases with increasing the distance from the BMU and current training epoch number (Davies, 2012). This process continues until some stopping criteria have been met (selected number of training epochs, for example).

---

1 Source: http://www.lohninger.com/kohonen.html
The classification algorithm is implemented in MATLAB and consists of the following steps:

1) A SOM is designed with specific parameters – number of units, map topology, training parameters, etc;

2) The training data subset is used for training of the SOM classifier;

3) A structured representation of the input space is obtained, for which the position of the SOM nodes and the distances between them are relevant to the configuration of the input training data vectors. In this way, nodes corresponding to similar texture samples are situated close to each other, as shown on the sample between-node distance map given in Figure 2-13;

4) Based on the available expert knowledge for the training samples, the count of the samples belonging to a certain class is determined for each map node;

5) Each node is then labelled to represent just one class - the one with highest value of an indicator \( L_i \), received by taking into account the classes hit histogram for that node \( h_i \) (i.e. the number of samples from each class mapped to that particular node) and a weighted (based on the learned distance between the nodes - \( w_{ij} \)) contribution of its direct neighbour classes hit histograms \( h_j \), as given by the following equation:

---

\[ L_i = \frac{h_i}{\sum_j w_{ij} h_j} \]

---

2 Source: [http://www.mclid.co.uk/](http://www.mclid.co.uk/)
\[ L_i = h_i + \sum_{j \in N} w_{ij} h_j, \]  

(2.13)

where \( N \) is the number of direct neighbours of the investigated node. Sample labelling map is given in Figure 2-12, where the class labels are depicted in different colours;

6) Classifier’s performance is tested on the remaining 30% of the available data and its classification accuracy rate is calculated according to:

\[ \text{Accuracy} = \frac{n_c}{n_m + n_c} \cdot 100 \% \],  

(2.14)

where \( n_m \) and \( n_c \) are the number of the misclassified and correctly classified samples, respectively;

7) Additional useful statistical measures (confusion matrices, precision, recall, etc.) are also recorded and saved alongside with the trained SOM classifiers for further investigation and analysis.

### 2.5 Results

In this section, several experiments exploring different linear and non-linear methods for dimensionality reduction by suppressing redundant information in the data are evaluated and compared. They are employed to transform the initially extracted 34-dimensional features to up to 7-dimensional data that is then used to train SOM classifiers, according to the proposed in section 2.4 algorithm.

Evaluation is then conducted in terms of comparing the separability of the classes in the transformed low-dimensional data distribution and by calculating the average classification accuracies over a testing set. In particular, the following methods are considered and compared during this investigation:
linear PCA using covariance matrix, linear PCA using correlation matrix, Kernel PCA, Hierarchical NLPCA, and Symmetrical NLPCA using linear and non-linear nodes in the bottleneck layer.

For each of studied approaches, the number of the extracted features is varied from 1 to 7 in order to evaluate their ability for redundancy suppression. In each of the experiments, 70% of the available data is used to train the SOMs, while the remaining 30% is used for testing. The mean percentages of the successfully classified texture images from the testing set are recorded over 15 runs of the SOM training procedure. For the sake of uniformity, after a number of simulations and evaluations (not presented here for keeping the chapter concise, but reported in (Petrov, Georgieva, & Jordanov, 2013)) all SOMs are chosen to have 15x8 topologies and are trained for 150 epochs.

The initial analysis of the input data, presented in Figure 2-6, illustrates the within- and between class distributions using 3 of the 34 features of the original data – the 1st Co-occurrence matrix feature, the 1st Laws’ mask feature and the Entropy feature. As it can be seen from the figure, the information contained in these 3 features is not sufficient to confidently discriminate between any of the classes present in the data. Although some of the classes are compactly clustered in specific locations of the plot (e.g., Speckled, Lisbon and partially Pebble), these locations overlap with clusters of other classes. Similar class distribution observations can be made by selecting any 3 of the original 34 extracted features, which is expected to lead to unsatisfactory classification results if only feature selection is used as a technique for dimensionality reduction and no feature transformation is implemented. In this context, the future investigation focuses on applying and evaluating different feature extraction techniques in the pre-processing stage of the proposed algorithm.
The first two experiments start by assessing the use of two linear feature extraction techniques – PCA, implemented in terms of data covariance matrix and PCA implemented in terms of data correlation matrix, respectively. A visual representation of the received inter- and intra-class variance distribution is presented in Figure 2-7 with respect to the first 3 principal components (PCs) for the better performing technique (the one using correlation matrix, according to the results, presented in Table 2-1). As it can be seen, three out of seven classes (Beach, Corkstone and Desert) are reasonably easy to be separated, whereas the rest of them considerably overlap each other, leading to unsatisfactory classification results (Table 2-1).
Next, as described in section 2.3, a Kernel PCA (KPCA) is incorporated in the proposed algorithm, mainly for comparison purposes. The resultant class distributions are presented in Figure 2-8a. Although some non-linearity is introduced in the inter- and intra-class variance and the overall separability is improved, still the classifier is only able to satisfactory distinguish between four out of the seven classes (Beach, Corkstone, Desert and Pebble). The latter can be confirmed by the classification results presented in Table 2-1 and the confusion matrix given in Figure 2-14a, as well as by the insignificant distances between the classes’ means of the misclassified classes, as given with 95% confidence intervals in Figure 2-8b.

![Figure 2-8. Within- and between-class variance, according to the first 3 KPCA components. (a) Scatter plot of the transformed data samples from the seven classes. (b) Texture classes means with 95% confidence intervals.](image-url)
In contrast, by including the presented in section 2.3 NLPCA feature extraction technique, the transformation results appear to be significantly improved. The inter-class separability is strengthened and the classes’ means with 95% confidence intervals are further apart, facilitating the classifier’s operation. The corresponding accuracy rates are summarised in Table 2-1 and sample results for a Symmetrical NLPCA run, with non-linear nodes in the bottleneck layer, are given in Figure 2-9a and Figure 2-9b.

Furthermore, despite the fact that the NLPCA is an entirely unsupervised dimensionality reduction technique, it is still capable to represent to a great extent the information from the initially extracted 34 textural features, even with just two components. This is demonstrated in Figure 2-10, where four (Lisbon, Precision, Pebble and Corkstone) out of the seven classes can be clearly discriminated and the remaining three (Beach, Desert, and Speckled) have very slight overlapping on the cluster edges.

Figure 2-9. Within- and between-class variance, considering the first 3 NLPCA components. (a) Scatter plot of the transformed data samples from the seven classes. (b) Texture classes means with 95% confidence intervals.
By analysing the summary results presented in Table 2-1, several major observations can be made. Firstly, the NLPCA techniques significantly outperform the linear PCA and KPCA algorithms in being able to preserve the valuable information of the initial 34 dimensional feature set, when transforming it to a very low dimensional (up to 3 components) input to a classifier. For instance, for both the linear PCA and KPCA techniques, the mean accuracy rates achieved when using 1 PC are about 47%, whereas NLPCA methods allow rates of up to 57.8%. Similarly, for 2 and 3 PCs, the investigated NLPCA approach gives superior results by at least 6%. Secondly, for more than 3 PCs, the performance of the NLPCA starts to get comparable to the one of linear PCA, but still outperforming the KPCA approach by about 4%. This saturation point is further investigated and discussed later in this section. Finally, the accuracy rates for the linear PCA start low, but gradually increase.
to a \textit{plateau} when about 7 PC are used, which is not the case for the non-linear algorithms considered. For both – the KPCA and NLPCA runs – interesting inversions of the trend of the accuracy rates are observed when increasing the number of used PCs. For example, the mean accuracy of the classifier drops slightly from 72.5\% to 71.8\% when the 5\textsuperscript{th} PC is added in the KPCA experiment.

A similar phenomenon is present in the NLPCA experiments, where adding the 4\textsuperscript{th} PC deteriorates the results by up to 6\%. Still, another inversion point is observed after adding the 6\textsuperscript{th} PC in the NLPCA case with non-linear activation function of the bottleneck layer.

<table>
<thead>
<tr>
<th></th>
<th>1 PC</th>
<th>2 PCs</th>
<th>3 PCs</th>
<th>4 PCs</th>
<th>5 PCs</th>
<th>6 PCs</th>
<th>7 PCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA Covariance</td>
<td>18.6</td>
<td>28.6</td>
<td>28.8</td>
<td>29.0</td>
<td>34.2</td>
<td>34.6</td>
<td>37.2</td>
</tr>
<tr>
<td>PCA Correlation</td>
<td>46.8</td>
<td>62.8</td>
<td>70.6</td>
<td>78.6</td>
<td>78.4</td>
<td>80.4</td>
<td>81.4</td>
</tr>
<tr>
<td>KPCA</td>
<td>Hierarchical</td>
<td>47.4</td>
<td>67.1</td>
<td>68.4</td>
<td>72.5</td>
<td>71.8</td>
<td>74.3</td>
</tr>
<tr>
<td>NLPCA Linear Activation Function</td>
<td>Symmetrical</td>
<td>54.6</td>
<td>73.2</td>
<td>77.6</td>
<td>76.2</td>
<td>77.6</td>
<td>78.2</td>
</tr>
<tr>
<td>NLPCA Nonlinear Activation Function</td>
<td>Symmetrical</td>
<td>57.8</td>
<td>76.0</td>
<td>78.6</td>
<td>72.4</td>
<td>84.6</td>
<td>80.6</td>
</tr>
</tbody>
</table>

The initial analysis of this inversion phenomenon shows that it can be attributed to the unsupervised nature of the investigated learning algorithm, where in this context some of the extracted PCs do not contribute towards the between-class separability, but capture side effects, such as illumination patterns in the data set. Another area for further investigation of this discrepancy is to be found in the 34 initially extracted features. Some of the misclassified samples suggest that the co-occurrence matrix features calculated for the neighbouring pixels and those with a distance of 5 might not be sufficient for robust capturing of some of the texture patterns presented in the data. To illustrate this suggestion, the samples shown in Figure 2-11 show images that are clustered in the same cell of a trained SOM classifier. Although they visually look different, their transformed feature vectors (given for 3 PC
as: 

\[
\begin{bmatrix}
-0.8343 & -0.2117 & 0.8857 \\
-0.8536 & -0.2039 & 0.7453
\end{bmatrix}
\]

respectively) are actually very close in terms of Euclidean distance.

![Figure 2-11. Example of texture images classified wrongly in the same SOM node: (a) Corkstone sample and (b) Lisbon sample.](image)

Additional insights of the algorithm performance can be revealed from Figure 2-12, where sample class clusters, identified by SOM classifiers are given. The presented SOM maps are coloured according to the class labels that each cell node corresponds to, where the colours are kept consistent with the class colouring scheme used in Figure 2-6 to Figure 2-8. The integers given in the cells denote the number of samples from the testing set that fall within a particular cell.

As it can be seen from the sample figure for the KPCA case given in Figure 2-12a, some of the classes are not very clearly clustered on the learned SOM map. In particular, the Lisbon, Precision and Speckled classes are not undoubtedly positioned, which is in correspondence with the observations already made for Figure 2-8 and the overall classification results presented in Table 2-1. On the other hand, the NLPCA classes are clearly separated and the test sample hits are more compactly distributed (Figure 2-12b). Also, clear class cluster boundaries can be observed on the sample SOM between-nodes distance map given in Figure 2-13 for the NLPCA case with non-linear activation function in the bottleneck layer.
Figure 2-12. Sample self-organising map clustering: (a) after KPCA feature extraction and (b) after NLPCA feature extraction.

Figure 2-13. Sample SOM between-node distances map for NLPCA run with non-linear activation function in the bottleneck layer. Darker colours represent greater distances.
The KPCA and NLPCA classifiers performance over the investigated textural classes can be further assessed by comparing the two sample confusion matrices given in Figure 2-14. As it can be seen from the figure, the overall classification accuracy increases from 68% to 81% by employing the proposed NLPCA approach instead of using KPCA and in general, the performance results improve for all but two of the classes. In particular, the accuracy rates for class Speckled are significantly improved from 36.4% to a 100%. The same is
observed for Corkstone, where the accuracy is improved by 30% and only 2 samples from the testing set are misclassified. Slightly inferior performance is observed for the Beach and Pebble classes, where the number of the correctly classified test samples is reduced by 2 and 3, respectively. Again, superior results are observed for the Desert (84.2%) and Lisbon classes (100%), compared to the KPCA case (68.4% and 78.6%, respectively).

Finally, interesting observations can be made by introducing different colouring schema for the SOM cell labels, according to the confidence of the samples mapped to the cells associated with a particular class. This is illustrated in Figure 2-15, where instead of depicting the SOM nodes regarding their winning class colour, they are presented in a colour received as a mixture of the main class colours, weighted by the class indicator coefficients, received from equation (2.13). The plots clearly reaffirm the NLPCA techniques ability to provide more robust texture clusters in comparison to the KPCA method, some of which learned clusters are very fuzzy and hard to discriminate (e.g., Precision and Speckled).

A comparison of the achieved classification accuracies with those reported by other authors applying unsupervised techniques for classification of cork products (mainly cork stoppers) shows competitive and in many of the cases superior nature of achieved here results. For example, the five different methods applied in (Oliveira et al., 2013) to classify cork stoppers using three quality labels resulted in between 68.3% and 73.3% mean accuracy. Costa and Pereira’s (2006) system for intelligent inspection of seven types of cork stoppers achieved overall success rate of 46%-58%. Radeva et al. (2002) reported similar results – up to 46% success rate for the extracted PCA feature set, when classifying five types of cork stoppers with their intelligent visual inspection system. In a subsequent work with Vitria and Bressan (Vitria et al., 2007) they again reported success rate between 40% and 46% for the PCA case (although the mean result of their non-parametric classifiers was 45%, they received much
higher accuracy rate for the parametric ones (on the original data) – 90% for a Maximum Likelihood and 98% for a Naive Bayes approach with coefficient-constrained independent component analysis (CC-ICA)). High accuracy of above 90% is also reported in (J. Xie et al., 2015), however it is achieved by implementing a supervised technique.

Figure 2.15. Sample self-organising map clustering with colour blurring, reflecting the class probabilities for each neighbouring cell. (a) After KPCA feature extraction. (b) After NLPCA feature extraction.
2.6 Conclusions

Texture image classification of different cork tile samples has been investigated as a pattern recognition problem in this chapter. The functionality of an existing system has been extended with unsupervised learning capabilities and the main focus of the proposed approach has been kept on the enhanced use of nonlinear dimensionality reduction techniques for limiting the number of features (describing the texture image) used as input to a SOM classifier.

Two main nonlinear PCA approaches have been analysed, namely NLPCA and KPCA and the obtained results have been compared with those from a linear PCA. Applying the proposed nonlinear techniques led to improving the classification results (achieving mean accuracy of more than 84%), obtained using fewer components than the linear PCA.

The improvement introduced by using nonlinear approaches, however, cannot be completely attributed to the NLPCA itself, since the SOM classifier performs also a sort of nonlinear dimensionality reduction that narrows the gap between the obtained results with linear and nonlinear PCA. A further increase of the SOM accuracies can be achieved by refining the class boundaries, in order to avoid misclassification of samples having fine visual texture discrepancies.

Although a straightforward comparison of the method performance, based only on the classification accuracy, can be misleading due to the different complexity of the investigated problems (size and quality of the datasets, differences in the preprocessing and feature extraction techniques, training convergence parameters, etc.), it still can give some indication about the method quality when compared with other authors (Costa & Pereira, 2006; Egmont-Petersen et al., 2002; Oliveira et al., 2013; Radeva et al., 2002; Vitria et al., 2007) who use unsupervised classification approaches.
Three known techniques have been used at the feature extraction phase in this investigation - one from each of the three general groups: *Entropy* from the statistical approaches; *Co-occurrence matrices* from the geometrical (spatial) techniques; and *Laws’ masks* from the filter methods. Considering additional techniques for feature extraction (e.g., model based approaches) and enlarging the dataset size (since the number of samples used to derive the different NLPCA is relatively low and the associate AANNs are prone to overfitting, reducing in this way the abilities of the subsequent SOM classification), can be an area for future extension of this work, aiming to enhance the system and improve its overall classification results.

As a result of the research presented in this chapter, two journal ([2] and [4] in the journal publication list (Section 6.3)) and two peer-reviewed conference papers ([4] and [5] in the conference publication list) have been prepared for publication and presentation.
Chapter 3  Intelligent Radar Signal Recognition and Classification

This chapter addresses a classification task for timely and reliable identification of intercepted radar signals. The problem is to classify the signals by finding patterns in their pulse train characteristics (such as operational frequencies, modulation types, pulse repetition intervals, scanning period, etc.) into several functional groups. The main challenge is caused by the inherent considerable number of records containing missing values (i.e. values that have not been intercepted or recognised).

To solve the classification task, two separate approaches are investigated, implemented, tested and validated on a number of case studies. In the first approach, a listwise deletion is used to clean the data of samples containing missing values and then feed-forward neural networks are employed for the classification task. In the second one, a multiple imputation (MI) model-based method for dealing with missing data (by producing confidence intervals for unbiased estimates without loss of statistical power, i.e. by using all the available samples) is investigated. Afterwards, a feedforward backpropagation neural network is trained to solve the signal classification problem. Each of the approaches is tested and validated on a number of case studies and the results are evaluated and critically compared.

The rest of the chapter is organised as follows: the next section (3.1) presents a review of related literature and relevant background knowledge on the
investigated topic. Next, in section 3.2 a broader formulation of the problem is provided and a deeper analysis of the available data set is made. Different statistical transformation techniques are discussed and a multiple imputation method for dealing with missing data is introduced in section 3.3. Several NN topologies, training parameters, input and output coding, and data transformation techniques for facilitating the learning process are tested and evaluated on a set of case studies in section 3.4. Finally, section 3.5 summarises the results and provides ideas for further extension of this research.

3.1 Background Knowledge

Early radar technology had been secretly developed for military purposes in the late 1930s and since then it has found a wide range of applications. In the military area, RADARs (RAadar Detection And Ranging) are employed in detecting, locating, tracing, and identifying objects, for surveillance, navigation and weapon guidance purposes for terrestrial, marine, and air systems at small to medium and large distances (from ballistic missile defence systems to fist sized tactical missile seekers)(Richards, 2005).

For civilian purposes, the increasingly wide spread of applications includes: air traffic control systems used to guide commercial aircrafts in the vicinity of the airports and during their flight; airplanes use radars for determining altitude and avoiding severe weather locations; in sea navigation, radars are commonly used by ships for collision avoidance, and are now beginning to serve the same role for the automobile and trucking industries; police traffic radar are used for enforcing speed limits; airborne radars are used not only for weather forecast, large-scale weather monitoring, prediction and atmospheric research, but also for environmental monitoring of forestry conditions and land usage, water and ice conditions, pollution control, etc.; spaceborne (both satellite and space
shuttle) serve for space surveillance and planetary observation; in sport they are used for measuring the speed of tennis and baseball serves (Richards, 2005).

A basic block-scheme of a radar system is shown in Figure 3-1. Radars are considered to be “active” sensors, as they use their own source of illumination (a transmitter) for locating targets. The radars range, resolution and sensitivity are generally determined by their transmitter and waveform generator. Although the typical radar systems operate in the microwave region of the electromagnetic spectrum with frequency range of about 200 MHz to about 95 GHz (with corresponding wavelengths of 0.67m to 3.16mm), there are also radars that function at frequencies as low as 2 MHz and as high as 300 GHz (Skolnik, 2001).

![Figure 3-1. A block diagram of a basic radar system. Radars operate by transmitting electromagnetic energy toward targets and processing the observed echoes.](image)

The IEEE letter nomenclature for the common nominal radar bands is given in Table 3-1 (Engineers, 1976). The millimetre wave band is sometimes further decomposed into approximate sub-bands of 36 to 46 GHz (Q band), 46 to 56 GHz (V band), and 56 to 100 GHz (W band). The lower frequency bands are usually preferred for longer range surveillance applications due to the low atmospheric attenuation and high available power and vice versa, the higher frequencies tend to be used for shorter range applications and higher
resolution, due to the smaller achievable antenna beam widths for a given antenna size, higher attenuation, and lower available power (Richards, 2005).

Table 3-1. Letter Nomenclature for Nominal Radar Frequency Bands (IEEE, 1976).

<table>
<thead>
<tr>
<th>Band</th>
<th>Frequencies</th>
<th>Wavelengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>HF</td>
<td>3–30 MHz</td>
<td>100–10 m</td>
</tr>
<tr>
<td>VHF</td>
<td>30–300 MHz</td>
<td>10–1 m</td>
</tr>
<tr>
<td>UHF</td>
<td>300 MHz –1 GHz</td>
<td>1–0.3 m</td>
</tr>
<tr>
<td>L</td>
<td>1–2 GHz</td>
<td>0.3–0.15 m</td>
</tr>
<tr>
<td>S</td>
<td>2–4 GHz</td>
<td>15–7.5 cm</td>
</tr>
<tr>
<td>C</td>
<td>4–8 GHz</td>
<td>7.5–3.75 cm</td>
</tr>
<tr>
<td>X</td>
<td>8–12 GHz</td>
<td>3.75–2.5 cm</td>
</tr>
<tr>
<td>Ku</td>
<td>12–18 GHz</td>
<td>2.5–1.67 cm</td>
</tr>
<tr>
<td>K</td>
<td>18–27 GHz</td>
<td>16.7–11.1 mm</td>
</tr>
<tr>
<td>Ka</td>
<td>27–36 GHz</td>
<td>11.1–7.5 mm</td>
</tr>
<tr>
<td>Q,V,W</td>
<td>36–300 GHz</td>
<td>7.5–1 mm</td>
</tr>
</tbody>
</table>

Based on their characteristics, features and application areas, radars can be classified in terms of the following criteria (Capraro, Farina, Griffiths, & Wicks, 2006):

- **purpose and function**: surveillance, tracking, guidance, reconnaissance, imaging, data link;

- **frequency band**: radar systems have been operating at frequencies as low as 2 MHz and as high as 300 GHz (see Table 3-1). Criteria for frequency selection for surveillance radar can be found in (Galejs, 2000; Skolnik, 2001);

- **waveform**: continuous wave, pulsed wave, digital synthesis;

- **beam scanning**: fixed beam, mechanical scan (rotating, oscillating), mechanical scan in azimuth, electronic scan (phase control, frequency control and mixed in azimuth/elevation), mixed (electronic-mechanical) scan, multi-beam configuration;
- **location**: terrestrial (stable, mobile), marine-borne, air-borne, space-borne;

- **spectrum of collected data**: range (delay time of echo), azimuth (antennae beam pointing, amplitude of echoes), elevation (3D - radar, multifunctional, tracking), height (derived by range and elevation), intensity (echo power), radar cross section (RCS) - (derived by echo intensity and range), radial speed (measurement of differential phase along the time on target due to the Doppler effect - it requires a coherent radar), polarimetry (phase and amplitude of echo in the polarisation channels: horizontally transmitted - HH, horizontally received - HV, VH, VV), RCS profiles along range and azimuth (high resolution along range, imaging radar);

- **configuration**: monostatic (same antenna with co-located transmitter and receiver), bi-static (two antennas), multistatic (one or more spatially dispersed transmitters and receivers). Further detail on mono-, bi- and multistatic radars can be found in (Johnsen & Olsen, 2006);

- **signal processing**: coherent (Moving Target Detector / Pulse-Doppler / super-resolution signal processor / synthetic aperture radars (SAR)), non-coherent (integration of envelope signals, moving window, adaptive threshold (Constant False Alarm Rate (CFAR)) and mixed (Cumming & Wong, 2005);

- **transmitter and receiver technologies**: antenna - reflector plus feed, array (planar, conformal), corporate feed; transmitter - magnetron, klystron, wideband amplifiers (high-power travelling wave tubes (TWT)), solid state; and receiver - analogue and digital technologies, base band, intermediate frequency sampling, low-power TWT;

- **area of application**: large-scale weather forecast and monitoring, air traffic control and guidance (terminal area, en route, collision avoidance,
airport apron); police traffic radar used for enforcing speed limits; air
defence; anti-theatre ballistic missile defence; vessel traffic surveillance;
remote sensing (application to crop evaluation, geodesy, astronomy,
defence); environmental monitoring of forestry conditions and land
usage; pollution control; geology and archaeology (ground penetrating
radar); meteorology (hydrology, rain/hail measurement); study of
atmosphere (detection of micro-burst and gust, wind profilers); space-
borne altimetry for measurement of sea surface height; acquisition and
tracking of satellites; monitoring of space debris; marine - navigation
and ship collision avoidance; others.

Radar detection, classification and tracking of targets against a background of
clutter and interference are considered as “general radar problem”. For military
purposes, the general radar problem includes searching for, interception,
localisation, analysis and identification of radiated electromagnetic energy,
which is commonly known as radar Electronic Support Measures (ESM). They
are considered to be a reliable source of valuable information regarding threat
detection, threat avoidance, and, in general, situation awareness for timely
deployment of counter-measures (Schleher, 1999; Sciortino, 1997). A list of ESM abbreviations is given in Table 3-2.

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>EW</td>
<td>Electronic warfare</td>
</tr>
<tr>
<td>MOP</td>
<td>Modulation on pulse</td>
</tr>
<tr>
<td>PA</td>
<td>Pulse amplitude</td>
</tr>
<tr>
<td>PDW</td>
<td>Pulse descriptor word</td>
</tr>
<tr>
<td>PPI</td>
<td>Pulse-to-pulse interval</td>
</tr>
<tr>
<td>PRI</td>
<td>Pulse repetition interval</td>
</tr>
<tr>
<td>PD</td>
<td>Pulse duration</td>
</tr>
<tr>
<td>PW</td>
<td>Pulse width</td>
</tr>
<tr>
<td>RF</td>
<td>Radio frequency</td>
</tr>
<tr>
<td>TOA</td>
<td>Time of arrival</td>
</tr>
<tr>
<td>ST</td>
<td>Scanning type</td>
</tr>
<tr>
<td>SP</td>
<td>Scan period</td>
</tr>
</tbody>
</table>
A real-time identification of the radar emitter associated with each intercepted pulse train is a very important function of the radar ESM. Typical approaches include sorting incoming radar pulses into individual pulse trains, then comparing their characteristics with a library of parametric descriptions, in order to get list of likely radar types. This can be very difficult task as there may be radar modes for which there is no record in ESM library; overlaps of different radar type parameters; increases in environment density (e.g., Doppler spectrum radars, transmitting hundreds of thousands of pulses per second); agility of radar features, such as radio frequency and scan, pulse repetition interval, etc.; Also, multiplication and dispersion of the modes for military radars; noise and propagation distortion that lead to incomplete or erroneous signals (Granger, Rubin, Grossberg, & Lavoie, 2001).

3.1.1 Neural Networks in Radar Recognition Systems

There are wide variety of approaches and methods used for radar emitter recognition and identification. For example, (D’Agostino, Foglia, & Pistoia, 2009) investigates a Specific Emitter Identification technique applied to ESM data and by analysing the radar pulses tries to extract unique features for each radar, which can be later used for identification. In (L. Li, Ji, & Wang, 2009) wavelet packet transform is used for feature extraction and then probabilistic support vector machines SVMs are employed for the radar emitter recognition task. SVMs are also used in (Zhai & Jiang, 2015; G. Zhang, Hu, & Jin, 2004) for solving a similar problem. In (Lunden & Koivunen, 2007) the authors focus their research on the estimation of a common modulation from a group of intercepted radar pulses and use it as a basis for specific emitter identification. A variety of novel radar emitter recognition algorithms, incorporating clustering and competitive learning, and investigating their advantages over
the traditional methods are proposed in (Feng & Lin, 2014; Hassan, 2003; J. Li & Ying, 2014; J. Liu, Lee, Li, Luo, & Wong, 2005; Pang, Lin, & Xu, 2014; Ting, Wei, & Bing, 2011; L. Wang, Ji, & Shi, 2010; Zeng & Li, 2010).

Among those approaches, a considerable part of the research in the area incorporates NN, due to their parallel architecture, fault tolerance and ability to handle incomplete radar type descriptions and inconsistent and noisy data (Ripley, 2008). NN techniques have previously been applied to several aspects of radar ESM processing and radar type recognition (Sciortino, 1997), including Pulse Descriptor Word (PDW) sorting (Kamgar-Parsi, Kamgar-Parsi, & Sciortino Jr, 1996; Pape, Anderson, Carter, & Wasilousky, 1997). More recently, many new radar recognition systems include NNs as a key classifier (Anjaneyulu, Sarma, & Murthy, 2009; Ibrahim, Abdullah, & Saripan, 2009; Peipei & Hui, 2014; Yin, Yang, Yang, Zuo, & Gao, 2011; Z.-C. Zhang, Guan, & He, 2009). Some examples of NN architectures and topologies used for radar identification recognition and classification based on ESM data include Multilayer Perceptron (MLP) (Ripley, 2008), Radial Basis Function (RBF) (Chen, Cowan, & Grant, 1991), a vector neural network (Shieh & Lin, 2002), and single parameter dynamic search neural network (Yin et al., 2011).

In many cases, the NNs are hybridised with other techniques, including fuzzy systems, clustering algorithms, wavelet packets, or Kalman filters. When implementing their “What-and-Where fusion strategy” (Granger et al., 2001) use an initial clustering algorithm to separate pulses from different emitters according to position-specific parameters of the input pulse stream, and then apply fuzzy ARTMAP (based on Adaptive Resonance Theory (ART) neural network) to classify streams of pulses according to radar type, using their functional parameters. They also complete simulations with a data set that has missing input pattern components and missing training classes and then incorporate a bank of Kalman filters to demonstrate high-level performance of their system on incomplete, overlapping and complex radar data. In
(Anjaneyulu et al., 2009) higher order spectral analysis (HOSA) techniques are used to extract information from low probability of intercept (LPI) radar signals to produce 2D signatures, which are then fed to a NN classifier for detecting and identifying the LPI radar signal. The work presented in (Ibrahim et al., 2009) investigates the potential of NNs (MLPs) when used in Forward Scattering Radar (FSR) applications for target classification. The authors analyse collected radar signal data and extract features, which are then used to train NN for target classification. They also apply K-Nearest Neighbour classifier to compare the results from the two approaches and conclude that the NN solution is superior. In (Ting, Jingqing, & Bing, 2007) an approach combining rough sets (for data reduction) and NN as a classifier is proposed for radar emitter recognition problem, while (Azimi-Sadjadi, Yao, Huang, & Dobeck, 2000) combines wavelet packets and neural networks for target classification.

The common denominator of all referenced approaches is that they use supervised NN learning. This means that there is an available data set (or it is on-line collected), on which the NN can be trained and later used to determine the type of the radar emitters detected in the environment. During the training, the NN is presented with labelled samples from the available dataset and the NN weights are adjusted in order to minimise the difference between the NN output and the available target (supervised learning). This difference is expressed by an error function that is minimised towards the NN weights. One of the most popular methods for training is backpropagation (BP), however, as it uses Newton and queasy-Newton deterministic minimisation methods, it could become trapped in a local minimum and in this way to converge to a suboptimal training. Another drawback of the BP algorithm is that it can, sometimes, be slow and unstable. After training, the NN is tested for its ability to generalise, in other words, its ability to correctly classify samples that have not been shown during the learning process.
Among other considerations, the complexity of the training includes selecting the way of showing the samples to the network (i.e. how the training data set is organised and presented to the NN – ‘batch mode’, ‘on-line mode’, etc). Another important question is when to stop the training – achieving a zero error function does not always lead to an optimal training. The reality shows that at some point of the learning process, the NN starts to memorise rather than to generalise – this when the NN starts to overfit. In order to avoid this, an additional data subset (called validation subset), is used in parallel with the training set. Initially, the errors on both sets will decrease, but at some point the validation error will start to rise, while the training error will continue to decrease. This point is an indication of overfitting and the training should be stopped, with the current weights assumed to be optimal. This training approach is known as split sample training, where the available dataset is split in training, validation and testing subsets. There are also other training approaches, such as k-fold crossover, or bootstrapping, each with their own specific advantages and drawbacks (Ripley, 2008). One advantage of the k-fold crossover, for example, is that it can be applied when limited number of samples is available for training. In addition, often before approaching training, the available data set needs to be pre-processed. Radar signal processing has specific features that differentiate it from most other signal processing fields. Many modern radars are coherent, meaning that the received signal, once demodulated to baseband, is complex-valued rather than real-valued and as it can be seen from Table 3-4, many of the collected data is categorical. Another specificity of the radar data sets is that there are usually many missing or incomplete data. Therefore, the problems of representation and statistical pre-processing of the available dataset are very important steps that need to be considered, before starting the actual training. This may also include transformation techniques, such as linear discriminant analysis and principal component analysis, in order to reduce the dimensionality of the problem and dispose of redundant information in the dataset.
3.1.2 Dealing with Missing Data

According to statistical analysis, the nature of missing data can be classified in three main groups (Baraldi & Enders, 2010; Enders, 2010; Graham, 2009a): missing completely at random (MCAR), where the probability that an observation is missing is unrelated to its value or to the value of any other variables; missing at random (MAR) – that missingness does not depend on the value of the observed variable, but on the extent the missingness correlation with other variables that are included in the analysis (in other words, the cause of missingness is considered); and missing not at random (MNAR) – when the data are not MCAR or MAR (missingness still depends on unobserved data).

The problem associated with MNAR is that it yields biased parameter estimates, while MCAR and MAR analysis yield unbiased ones (at the same time the main consequence of using MCAR is loss of statistical power), (Graham, 2009b).

Dealing with missingness requires an analysis strategy leading to least biased estimates, while not losing statistical power. The problem is these criteria are contradictory and in order to use the information from the partial data in samples with missing data (keeping up the statistical power), and substituting the missing data samples with estimates, inevitably brings bias.

The most popular approaches in dealing with missing data generally fall in three groups: Deletion methods; Single imputation methods; and Model-based methods (Enders, 2010; Horton & Lipsitz, 2001; Little & Rubin, 2002).

- Deletion methods include pairwise and listwise deletion. The pairwise deletion (also called "unwise" deletion) keeps as many samples as possible for each analysis (and in this way uses all available information for it), resulting in incomparable analysis, as each is based on different subsets of data, with different sample sizes and different standard
errors. The listwise deletion (also known as complete case analysis) is a simple approach, in which all cases with missing data are omitted. The advantages of this technique include comparability across the analyses and it leads to unbiased parameter estimates (assuming the data is MCAR), while its main disadvantage is that there may be substantial loss of statistical power (because not all information is used in the analysis, especially if a large number of cases is excluded).

- The single imputation methods include mean/mode substitution, dummy variable method, and single regression. Mean/mode substitution is an old procedure, currently rejected due to of its intrinsic problems, e.g., it adds no new information (the overall mean stays the same), reduces the variability, and weakens the covariance and correlation estimates (it ignores relationship between variables). The dummy variable technique uses all available information about missing observation, but produces biased estimates. In the regression approach, linear regression is used to predict what the missing value should be (based of the available other variables) and then uses it as an actual value. The advantage of this technique is that it uses information from the observed data, but overestimates the model fit and the correlation estimates, and weakens the variance (Enders, 2010).

- Most popular, “modern” model-based approaches, fall into two categories: multiple imputation (MI) and maximum likelihood (ML) methods (often referred to as full-information maximum likelihood), (Graham, 2009a). Their advantage is that they model the missingness and give confidence intervals for estimates, rather than relying on a single imputation. If the assumption for MAR holds, both groups of methods result in unbiased estimates (i.e., tend to “preserve” means, variances, co-variances, correlations and linear regression coefficients) without loss of statistical power.
ML identifies a set of parameter values that produces the highest (log) likelihood and estimates the most likely value that would result in the observed data. It has the advantage that both complete and incomplete cases are used, in other words, it utilises of the full information and produces unbiased parameter estimates (with MCAR/MAR data). The MI approach involves three distinct steps: first, sets of plausible data for the missing observations are created and these sets are filled in separately to create many ‘completed’ datasets; second, each of these datasets is analysed using standard procedures for complete datasets; and thirdly, the results from previous step are combined and pooled into one estimate for the inference. The aim of the MI process is not just to fill in the missing values with plausible estimates, but also to plug in multiple times these values by preserving important characteristics of the whole dataset. As with most multiple regression prediction models, the danger of overfitting the data is real and can lead to less generalisable results than would have been possible with the original data (Osborne & Overbay, 2012).

The advantage of the MI technique is that it provides more accurate variability by making multiple imputations for each missing value (it considers both variability due to sampling and variability due to imputation) and its disadvantage is that it depends on the correctly specified model. Also, it requires cumbersome coding, but the latter is not an issue due to the existence of easy to use off-shelf software packages. For the purpose of this investigation, a free, open source R statistical software is used.
3.2 Data Analysis

For the purpose of this research, a data set composed of 29094 intercepted generic data samples is used. Each of the captured signals is pre-classified by experts in one of 26 categories, in regards to the platform that can carry the radar emitter (aircraft, ship, missile, etc.) and in one of 142 categories, based on the functions it can perform (3D surveillance, weather tracking, air traffic control, etc.).

Each data entry represents a list of 12 recorded pulse train characteristics (signal frequencies, modulation type, pulse repetition intervals, etc. that will be considered as input parameters), a category label (specifying the radar function and being treated as system output) and a data entry identifier (for reference purposes only) (Table 3-3).

Table 3-3. Sample radar data subset. Missing values (i.e. values that could not have been intercepted or recognised) are denoted by ‘-’. The rest of the acronyms are defined in A more comprehensive summary of the data distribution is presented in Table 3-4, where an overview of the type, range and percentage of missing values for the recorded signal characteristics is given. The collected data consists of both numerical (integer and float) and categorical values, therefore coding of the categorical fields to numerical representations will be required during the data pre-processing stage. Also, due to the large number of missing values for some of the parameters, approaches for handling of missing data will be considered.

<table>
<thead>
<tr>
<th>ID</th>
<th>FN</th>
<th>RFC</th>
<th>RFmin</th>
<th>RFmax</th>
<th>PRC</th>
<th>PRImin</th>
<th>PRImax</th>
<th>PDC</th>
<th>PDmin</th>
<th>PDmax</th>
<th>ST</th>
<th>SPmin</th>
<th>SPmax</th>
</tr>
</thead>
<tbody>
<tr>
<td>84</td>
<td>SS</td>
<td>B</td>
<td>5300</td>
<td>5800</td>
<td>K</td>
<td>-</td>
<td>-</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>A</td>
<td>5.9</td>
<td>6.1</td>
</tr>
<tr>
<td>4354</td>
<td>AT</td>
<td>F</td>
<td>2700</td>
<td>2900</td>
<td>F</td>
<td>1351.3</td>
<td>1428.6</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>A</td>
<td>9.5</td>
<td>10.5</td>
</tr>
<tr>
<td>7488</td>
<td>3D</td>
<td>B</td>
<td>8800</td>
<td>9300</td>
<td>K</td>
<td>100</td>
<td>125</td>
<td>S</td>
<td>13</td>
<td>21</td>
<td>B</td>
<td>1.4</td>
<td>1.6</td>
</tr>
<tr>
<td>9632</td>
<td>WT</td>
<td>F</td>
<td>137</td>
<td>139</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>V</td>
<td>-</td>
<td>-</td>
<td>D</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>9839</td>
<td>3D</td>
<td>S</td>
<td>2900</td>
<td>3100</td>
<td>J</td>
<td>-</td>
<td>-</td>
<td>V</td>
<td>99</td>
<td>101</td>
<td>A</td>
<td>9.5</td>
<td>10.5</td>
</tr>
</tbody>
</table>
A more comprehensive summary of the data distribution is presented in Table 3-4, where an overview of the type, range and percentage of missing values for the recorded signal characteristics is given. The collected data consists of both numerical (integer and float) and categorical values, therefore coding of the categorical fields to numerical representations will be required during the data pre-processing stage. Also, due to the large number of missing values for some of the parameters, approaches for handling of missing data will be considered.

**Table 3-4. Data description and percentage of missing values. In column “Type”: I – integer; C – categorical; R – real values.**

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Description</th>
<th>Type</th>
<th>Categories</th>
<th>Missing</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Reference for the line of data</td>
<td>I</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>FN</td>
<td>Function performed by the radar ('3D' – 3D surveillance, 'AT' – airtraffic control, 'SS' – surface search, 'WT' – weather tracker, etc.)</td>
<td>C</td>
<td>142</td>
<td>1.35%</td>
</tr>
<tr>
<td>RFC</td>
<td>Type of modulation used by the radar to change the frequency of the radar from pulse to pulse ('A' – agile, 'F' – fixed, etc.)</td>
<td>C</td>
<td>12</td>
<td>20.75%</td>
</tr>
<tr>
<td>Rfmin</td>
<td>Min frequency that can be used by the radar</td>
<td>R</td>
<td>-</td>
<td>11.15%</td>
</tr>
<tr>
<td>Rfmax</td>
<td>Max frequency that can be used by the radar</td>
<td>R</td>
<td>-</td>
<td>11.15%</td>
</tr>
<tr>
<td>PRC</td>
<td>Type of modulation used by the radar to change the Pulse Repetition Interval (PRI) of the radar from pulse to pulse ('F' – fixed, etc.)</td>
<td>C</td>
<td>15</td>
<td>15%</td>
</tr>
<tr>
<td>PRImin</td>
<td>Min PRI that can be used by the radar</td>
<td>R</td>
<td>-</td>
<td>46.70%</td>
</tr>
<tr>
<td>PRImax</td>
<td>Max PRI that can be used by the radar</td>
<td>R</td>
<td>-</td>
<td>46.70%</td>
</tr>
<tr>
<td>PDC</td>
<td>Type of modulation used by the radar to change the pulse duration of the radar from pulse to pulse ('S' – stable)</td>
<td>C</td>
<td>5</td>
<td>12.92%</td>
</tr>
<tr>
<td>PDmin</td>
<td>Min pulse duration that can be used by the radar</td>
<td>R</td>
<td>-</td>
<td>46.05%</td>
</tr>
<tr>
<td>PDmax</td>
<td>Max pulse duration that can be used by the radar</td>
<td>R</td>
<td>-</td>
<td>46.05%</td>
</tr>
<tr>
<td>ST</td>
<td>Scanning type – method that the radar uses to move the antenna beam ('A' – circular, 'B' – bidirectional, 'W' – electronically scanned, etc.)</td>
<td>C</td>
<td>28</td>
<td>11.33%</td>
</tr>
<tr>
<td>SPMn</td>
<td>Min scan period that can be used by the radar</td>
<td>R</td>
<td>-</td>
<td>59.35%</td>
</tr>
<tr>
<td>SPMx</td>
<td>Max scan period that can be used by the radar</td>
<td>R</td>
<td>-</td>
<td>59.35%</td>
</tr>
</tbody>
</table>
3.3 Data Pre-Processing

The pre-processing of the available data is of a great importance for the subsequent machine learning stage and usually can significantly affect the overall success or failure of the application of a given classification algorithm. In this context, the main objective of this stage is to analyse the available data for inconsistencies, outliers and irrelevant entries and to transform it in a form that could facilitate the underlying mathematical apparatus of the machine learning algorithm and lead to an overall improvement of the classifier’s performance.

3.3.1 Data Cleaning and Imputation

Data cleaning (also known as data cleansing or scrubbing) deals with detecting and removing errors and inconsistencies from data, in order to improve its quality (Rahm & Do, 2000). The most important tasks carried out on this stage would include identification of outliers (entries that are significantly different from the rest and could be a result of an error), resolving of data inconsistencies (values that are not consistent with the specifications or contradict expert knowledge), dealing with missing data (removing the missing values, assigning those values to the attributes’ mean, using statistical algorithms to predict the missing values) or removing redundant data in different representations.

At this stage of the pre-processing phase, two data sets are prepared. For the purposes of the first two case studies (presented later in this chapter), a data set only containing samples with complete data values is extracted, with the data that could not have been fully intercepted and recognised removed by applying
listwise deletion. The second data set (used for the final case study) is received after applying multiple imputation, performed as described below.

DEALING WITH MISSING DATA - DATA IMPUTATION

To estimate the values of the missing multivariate data, a sequential imputation algorithm, presented in (Verboven, Branden, & Goos, 2007) is used. According to it, if the available data set is denoted with $Y$ and the complete subset with $Y_c$, the procedure starts from the complete subset to estimate sequentially the missing values of an incomplete observation $Y^*$, by minimizing the covariance of the augmented data matrix $Y^* = [Y_c, x^*]$. Subsequently the data sample $x^*$ is added to the complete data subset and the algorithm continues with the estimate of next data sample with missing values.

Implementations in R of the original algorithm (available under the function name “impSeq”) and two modifications of it (namely “impSeqRob” and “impNorm”) are considered and tested. As the original algorithm uses the sample mean and covariance matrix, it is vulnerable to the presence of outliers, but this can be enhanced by including robust estimators of location and scatter (which is realised in the “impSeqRob” function). However, the outlyingness metric can be computed for a complete dataset only, therefore the sequential imputation of the missing data is done first and then the outlyingness measure is computed and used to define whether the observation is an outlier or not. If the measure does not exceed a predefined threshold, the observation is included in the next stage of the algorithm. In our investigation, however, the use of modified “impSeqRob” and “impNorm” versions did not produce better results when tested on complete dataset (which may be simply due to the lack of outliers), so the “impSeq” function was adopted.

After employing MI on the data samples with missing continuous values, a second dataset of 15656 observations is received, which is more than double the
size of the first dataset. Table 3-5 shows the inputted values produced by the MI algorithm for the sample subset, presented previously in Table 3-3.

**Table 3-5. Sample radar data subset with imputed values for the missing data entries.**

<table>
<thead>
<tr>
<th>ID</th>
<th>FN</th>
<th>RFC</th>
<th>RFmin</th>
<th>RFmax</th>
<th>PRC</th>
<th>PR1min</th>
<th>PR1max</th>
<th>PDC</th>
<th>PDmin</th>
<th>PDmax</th>
<th>ST</th>
<th>SPmin</th>
<th>SPmax</th>
</tr>
</thead>
<tbody>
<tr>
<td>84</td>
<td>SS</td>
<td>B</td>
<td>5300</td>
<td>5800</td>
<td>K</td>
<td>963.2</td>
<td>5625</td>
<td>S</td>
<td>5.8</td>
<td>17</td>
<td>A</td>
<td>5.9</td>
<td>6.1</td>
</tr>
<tr>
<td>4354</td>
<td>AT</td>
<td>F</td>
<td>2700</td>
<td>2900</td>
<td>F</td>
<td>1351</td>
<td>1428</td>
<td>S</td>
<td>4</td>
<td>6.3</td>
<td>A</td>
<td>9.5</td>
<td>10.5</td>
</tr>
<tr>
<td>7488</td>
<td>3D</td>
<td>B</td>
<td>8800</td>
<td>9300</td>
<td>K</td>
<td>100</td>
<td>125</td>
<td>S</td>
<td>13</td>
<td>21</td>
<td>B</td>
<td>1.4</td>
<td>1.6</td>
</tr>
<tr>
<td>9632</td>
<td>WT</td>
<td>F</td>
<td>137</td>
<td>139</td>
<td>T</td>
<td>622.6</td>
<td>31312</td>
<td>V</td>
<td>61.1</td>
<td>93.1</td>
<td>D</td>
<td>12</td>
<td>47.8</td>
</tr>
<tr>
<td>9839</td>
<td>3D</td>
<td>S</td>
<td>2900</td>
<td>3100</td>
<td>J</td>
<td>2058</td>
<td>48128</td>
<td>V</td>
<td>99</td>
<td>101</td>
<td>A</td>
<td>9.5</td>
<td>10.5</td>
</tr>
</tbody>
</table>

### 3.3.2 Data Coding and Transformation

This stage of the pre-processing aims to transform the data into a form that is appropriate for feeding to the selected classifier and would facilitate faster and more accurate machine learning. In particular, a transformation known as coding is applied to convert the categorical values presented in the data set to numerical ones. Three of the most broadly applied coding techniques are investigated and evaluated – continuous, binary and introduction of dummy variables.

For the first type of coding, each of the categorical values is substituted by a natural number, e.g., the 12 categories for the RFC input are encoded with 12 ordinal numbers, the 15 PRC categories – with 15 ordinal numbers, etc. A sample of data subset coded with continuous values is given in Table 3-6.

**Table 3-6. Sample subset with imputed radar data and natural number coding for the ‘RFC’, ‘PRC’, ‘PDC’, and ‘ST’ signal characteristics.**

<table>
<thead>
<tr>
<th>ID</th>
<th>RFC</th>
<th>RFmin</th>
<th>RFmax</th>
<th>PRC</th>
<th>PR1min</th>
<th>PR1max</th>
<th>PDC</th>
<th>PDmin</th>
<th>PDmax</th>
<th>ST</th>
<th>SPmin</th>
<th>SPmax</th>
</tr>
</thead>
<tbody>
<tr>
<td>84</td>
<td>2</td>
<td>5300</td>
<td>5800</td>
<td>7</td>
<td>963.2</td>
<td>5625</td>
<td>1</td>
<td>5.8</td>
<td>17</td>
<td>1</td>
<td>5.9</td>
<td>6.1</td>
</tr>
<tr>
<td>4354</td>
<td>4</td>
<td>2700</td>
<td>2900</td>
<td>4</td>
<td>1351</td>
<td>1428</td>
<td>1</td>
<td>4</td>
<td>6.3</td>
<td>1</td>
<td>9.5</td>
<td>10.5</td>
</tr>
<tr>
<td>7488</td>
<td>2</td>
<td>8800</td>
<td>9300</td>
<td>7</td>
<td>100</td>
<td>125</td>
<td>1</td>
<td>13</td>
<td>21</td>
<td>2</td>
<td>1.4</td>
<td>1.6</td>
</tr>
<tr>
<td>9632</td>
<td>4</td>
<td>137</td>
<td>139</td>
<td>11</td>
<td>622.6</td>
<td>31312</td>
<td>2</td>
<td>61.1</td>
<td>93.1</td>
<td>4</td>
<td>12</td>
<td>47.8</td>
</tr>
<tr>
<td>9839</td>
<td>9</td>
<td>2900</td>
<td>3100</td>
<td>6</td>
<td>2058</td>
<td>48128</td>
<td>2</td>
<td>99</td>
<td>101</td>
<td>1</td>
<td>9.5</td>
<td>10.5</td>
</tr>
</tbody>
</table>
Binary coding, wherein each non-numerical value is substituted by \( \log_2 N \) (where \( N \) is the number of categories taken by that variable) new binary variables (i.e. taking value of either 0 or 1), is illustrated in Table 3-7 for 32 categories.

<table>
<thead>
<tr>
<th>Index</th>
<th>Original Category</th>
<th>Encoded Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>B1</td>
</tr>
<tr>
<td>1</td>
<td>‘2D’</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>‘3D’</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>‘AA’</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>‘CS’</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>‘ME’</td>
<td>1</td>
</tr>
</tbody>
</table>

Finally, the non-numerical attributes are coded using dummy variables. In particular, every \( N \) levels of a categorical variable are represented by introducing \( N \) dummy variables. An example of dummy coding for 32 categorical levels is shown in Table 3-8.

<table>
<thead>
<tr>
<th>Index</th>
<th>Original Category</th>
<th>Encoded Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>D1</td>
</tr>
<tr>
<td>1</td>
<td>‘2D’</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>‘3D’</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>‘AA’</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>‘CS’</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>‘ME’</td>
<td>0</td>
</tr>
</tbody>
</table>

Taking into account the large number of categories presented for the categorical attributes in the input data set (Table 3-3), continuous and binary codings are considered for transforming the input variables. On the other hand, binary and dummy variable codings are chosen for representing the output parameters.
Finally, in order to balance the impact of the different input parameters on the training algorithm, data scaling is used. Correspondingly, each of the conducted experiments in this chapter is evaluated using 3 forms of the input data set: the original data (with no scaling); normalised data (i.e. scaling the attribute values within (0, 1) interval); and standardised data (i.e. scaling the attribute values to a zero mean and unit variance). A sample binary coded and standardised data subset is given in Table 3-9.

<table>
<thead>
<tr>
<th>ID</th>
<th>RFC</th>
<th>RFmin</th>
<th>RFmax</th>
<th>PRC</th>
<th>PRIm</th>
<th>PRImax</th>
<th>PDC</th>
<th>PDmin</th>
<th>PDmax</th>
<th>ST</th>
<th>SPmin</th>
<th>SPmax</th>
</tr>
</thead>
<tbody>
<tr>
<td>84</td>
<td>0001</td>
<td>5300</td>
<td>5800</td>
<td>0110</td>
<td>963</td>
<td>5625</td>
<td>0</td>
<td>5.8</td>
<td>17</td>
<td>00000</td>
<td>5.9</td>
<td>6.1</td>
</tr>
<tr>
<td>4354</td>
<td>0011</td>
<td>2700</td>
<td>2900</td>
<td>0011</td>
<td>1351</td>
<td>1428</td>
<td>0</td>
<td>4</td>
<td>6.3</td>
<td>00000</td>
<td>9.5</td>
<td>10.5</td>
</tr>
<tr>
<td>7488</td>
<td>0001</td>
<td>8800</td>
<td>9300</td>
<td>0110</td>
<td>100</td>
<td>125</td>
<td>0</td>
<td>13</td>
<td>21</td>
<td>00001</td>
<td>1.4</td>
<td>1.6</td>
</tr>
<tr>
<td>9632</td>
<td>0011</td>
<td>137</td>
<td>139</td>
<td>1010</td>
<td>622</td>
<td>31312</td>
<td>1</td>
<td>61.1</td>
<td>93.1</td>
<td>00011</td>
<td>12</td>
<td>47.8</td>
</tr>
<tr>
<td>9839</td>
<td>1000</td>
<td>2900</td>
<td>3100</td>
<td>0101</td>
<td>2058</td>
<td>48128</td>
<td>1</td>
<td>99</td>
<td>101</td>
<td>00000</td>
<td>9.5</td>
<td>10.5</td>
</tr>
</tbody>
</table>

### 3.3.3 System Training

The investigated neural network topologies include one hidden layer, with fully connected neurons in the adjacent layers and batch-mode training. For a given experiment with \( P \) learning samples, the error function is presented as:

\[
E_p = \frac{1}{2} \sum_{p=1}^{P} \sum_{i=1}^{L} (x_i^p - t_i^p)^2,
\]

where for each sample \( p = 1, \ldots, P \) and each neuron of the output layer \( i = 1, \ldots, L \), a pair \((x_i, t_i)\) of NN output and target values, respectively, is defined.

### 3.4 Results

A number of experiments are designed, implemented, executed and evaluated to test and validate the performance of the proposed intelligent system for
identification and classification of radar signals. Two separate approaches are considered and the related results are grouped and presented in the following two case studies. MATLAB® and its Statistics, Neural Networks and Global Optimisation toolboxes are used for coding and running of all the experiments.

### 3.4.1 Case Study 1 – Listwise Deletion and Feedforward Neural Networks

For the purposes of the first case study, samples that contain incomplete data (i.e. data that could not have been fully intercepted or recognised) are removed from the considered data set and as a result, a total of 7693 complete data samples of radar signal values are received. Subsequently, depending on the experiment to be performed in this case study, the samples are sorted by experts in several groups of major interest (according to their application). In 2 classes for the first two experiments ("Civil" and "Military") and in 11 classes for the purpose of the final one (4 from the “Civil” and 7 from the “Military” application areas).

A randomly selected, no missing data sample subset (after listwise deletion) is presented in Table 3-10. Its first column (the ID attribute) is retained for referencing purposes only and it is not used during the classifier’s training.

<table>
<thead>
<tr>
<th>ID</th>
<th>FN</th>
<th>RFC F</th>
<th>RFmin</th>
<th>RFmax</th>
<th>PRC</th>
<th>PR1min</th>
<th>PR1max</th>
<th>PDC</th>
<th>PDmin</th>
<th>PDmax</th>
<th>ST</th>
<th>SPmin</th>
<th>SPmax</th>
</tr>
</thead>
<tbody>
<tr>
<td>983</td>
<td>AT</td>
<td>15700</td>
<td>17700</td>
<td>F</td>
<td>100</td>
<td>142.9</td>
<td>S</td>
<td>0.03</td>
<td>0.05</td>
<td>2</td>
<td>A</td>
<td>0.9</td>
<td>1.1</td>
</tr>
<tr>
<td>1286</td>
<td>SS</td>
<td>5500</td>
<td>5800</td>
<td>K</td>
<td>909.1</td>
<td>1111.1</td>
<td>S</td>
<td>0.6</td>
<td>0.8</td>
<td>1.9</td>
<td>G</td>
<td>28</td>
<td>32</td>
</tr>
<tr>
<td>4846</td>
<td>SS</td>
<td>172</td>
<td>180</td>
<td>F</td>
<td>2439</td>
<td>2564.1</td>
<td>S</td>
<td>1.6</td>
<td>1.8</td>
<td>2439</td>
<td>32</td>
<td>28</td>
<td>32</td>
</tr>
<tr>
<td>12097</td>
<td>3D</td>
<td>5250</td>
<td>5850</td>
<td>F</td>
<td>2703</td>
<td>2777.8</td>
<td>S</td>
<td>3</td>
<td>3.3</td>
<td>5.8</td>
<td>6.2</td>
<td>12</td>
<td>60</td>
</tr>
<tr>
<td>28059</td>
<td>WM</td>
<td>5300</td>
<td>5700</td>
<td>F</td>
<td>1127</td>
<td>1132.5</td>
<td>S</td>
<td>0.75</td>
<td>0.85</td>
<td>12</td>
<td>60</td>
<td>12</td>
<td>60</td>
</tr>
</tbody>
</table>
Next, a coding transformation (as described in Section 3.3.2) is applied to convert the categorical values in the data set to numerical ones. Taking into account the large number of categories in the inputs (Table 3-3), continuous and binary codings are considered for transforming the input variables. On the other hand, binary and dummy variable representations are used for transforming the output parameters.

In order to balance the impact of the different input parameters on the training algorithm, data scaling is applied. Respectively, each of the experiments conducted for the purposes of this case study is evaluated using three forms of the input data set – the data itself (with no scaling), after normalisation (i.e. scaling the attribute values to fall within a specific range, for example [0 1]), and after standardisation (i.e. scaling the attribute values to a zero mean and unit variance). A sample binary coded and normalised data subset is given in Table 3-11.

<table>
<thead>
<tr>
<th>ID</th>
<th>RFC Enc.</th>
<th>RFCmin</th>
<th>RFCmax</th>
<th>PRC Enc.</th>
<th>PRImin</th>
<th>PRImax</th>
<th>PDC Enc.</th>
<th>PDbmin</th>
<th>PDmax</th>
<th>ST Enc.</th>
<th>STmin</th>
<th>STmax</th>
</tr>
</thead>
<tbody>
<tr>
<td>983</td>
<td>0 0 1 1</td>
<td>0.228</td>
<td>0.249</td>
<td>0 0 1 1</td>
<td>0.0006</td>
<td>0.0003</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1286</td>
<td>0 0 0 0</td>
<td>0.080</td>
<td>0.082</td>
<td>0 1 1 0</td>
<td>0.0056</td>
<td>0.0022</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4846</td>
<td>0 0 1 1</td>
<td>0.002</td>
<td>0.003</td>
<td>0 0 1 1</td>
<td>0.0151</td>
<td>0.0051</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12097</td>
<td>0 0 0 0</td>
<td>0.076</td>
<td>0.082</td>
<td>0 0 1 1</td>
<td>0.0167</td>
<td>0.0055</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>28059</td>
<td>0 0 1 1</td>
<td>0.077</td>
<td>0.080</td>
<td>0 0 1 1</td>
<td>0.0070</td>
<td>0.0023</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The investigated NN topologies include one hidden layer, with fully connected neurons in the adjacent layers and batch-mode training. For a given experiment with $P$ learning samples, the error function is given with (3.1). Supervised NN learning with Levenberg-Marquardt algorithm and tangent sigmoid transfer function (Figure 3-2) is used. A split-sample technique using randomly selected 70% of the available data for training, 15% for validation and 15% for testing, and mean squared error (MSE) is used for evaluating the learning performance. The stopping criteria is set to 500 training epochs, gradient reaching less than $1.0\times10^{-6}$ or if 6 consequent validation checks fail, whichever occurs first.
For the purposes of the first experiment, the categorical attributes of the input data are coded with consecutive integers. In this way a total of 12 input variables are received (Table 3-6). Two neural network topologies are examined – 12-10-1 (12 neurons in the input, 10 neurons in the hidden and 1 neuron in the output layers) and 12-10-2, where the output parameter is coded as one binary neuron taking values 0 (“Civil”) and 1 (“Military”) for the first topology and 2 binary neurons, taking values 10 (“Civil”) and 01 (“Military”) for the second topology (Figure 3-3).

Figure 3-2. Tangent sigmoid transfer function\(^3\).

Figure 3-3. Investigated NN topologies for Case Study 1: 12 neurons in the input layer, 10 in the hidden and 1 (a) or 2 (b) neurons in the output layer.

\(^3\)Source: http://www.mathworks.com/
The performance of each of the topologies is investigated, evaluated and compared after training with the original, normalised and standardised data. The results are summarised in Table 3-12.

<table>
<thead>
<tr>
<th>NN Topology</th>
<th>Inputs Scaling</th>
<th>Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-10-1</td>
<td>no scaling</td>
<td>78.12 %</td>
</tr>
<tr>
<td></td>
<td>normalisation</td>
<td>80.82 %</td>
</tr>
<tr>
<td></td>
<td>standardisation</td>
<td>80.76 %</td>
</tr>
<tr>
<td>12-10-2</td>
<td>no scaling</td>
<td>80.14 %</td>
</tr>
<tr>
<td></td>
<td>normalisation</td>
<td>81.60 %</td>
</tr>
<tr>
<td></td>
<td>standardisation</td>
<td>82.18 %</td>
</tr>
</tbody>
</table>

Sample confusion matrices for a 12-10-2 NN classifier trained with normalised input data and a validation stop activated after 118 epochs are given in Figure 3-4. They show the classifier’s performance on the training, validation, and testing subsets and its overall performance. The system demonstrates high accuracy, as shown by the high number of correct responses in the green squares and the low number of incorrect responses in the red squares. The lower right squares illustrate the overall classifier accuracies, which for the testing set is 81.6 %.
Figure 3-4. Classification results for 12-10-2 NN classifier with normalised input data and a validation stop after 118 epochs. The values in green specify the correctly classified samples for each class (10 - Civil, 01 - Military).

The second experiment investigates two additional NN topologies: 22-22-1 and 22-22-2, where the output parameter is again coded by one binary neuron (0 for “Civil” and 1 for “Military”) for the first topology and by two binary neurons for the second one (01 for “Civil” and 01 for “Military”). Again, the performance of each of the topologies is investigated, evaluated and compared using the original data, after normalisation and after standardisation. The performance results are summarised in Table 3-13.

<table>
<thead>
<tr>
<th>NN Topology</th>
<th>Inputs Scaling</th>
<th>Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>22-22-1</td>
<td>no scaling</td>
<td>81.90 %</td>
</tr>
<tr>
<td></td>
<td>normalisation</td>
<td>83.34 %</td>
</tr>
<tr>
<td></td>
<td>standardisation</td>
<td>83.01 %</td>
</tr>
<tr>
<td>22-22-2</td>
<td>no scaling</td>
<td>81.77 %</td>
</tr>
<tr>
<td></td>
<td>normalisation</td>
<td>83.90 %</td>
</tr>
<tr>
<td></td>
<td>standardisation</td>
<td>84.30 %</td>
</tr>
</tbody>
</table>
Similarly to the first experiment, a sample confusion matrix is presented in Figure 3-5 for a 22-22-2 NN classifier trained with standardised input data. A very high accuracy of 84.3% on the testing data set is achieved after 114 epochs and activation of the validation check stopping criteria (unsatisfactory performance on the validation data set in six successive iterations).

The final experiment in this case study investigates a broader output space of 11 classes (4 from the “Civil” and 7 from the “Military” domain) and evaluates a 22-22-11 NN classifier with unscaled, normalised and standardised training data using dummy variable coded outputs. Summary of the obtained results is presented in Table 3-14 and a sample confusion matrix for the investigated classifier with standardised input training data is given in Figure 3-6, where a good recognition rate of 67.49 % can be observed.
Table 3-14. Classification performance (over the testing set) for binary input coding and 22-22-11 topology with no data scaling, after normalisation and after standardisation.

<table>
<thead>
<tr>
<th>NN Topology</th>
<th>Inputs Scaling</th>
<th>Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>22-22-11</td>
<td>normalisation</td>
<td>66.70 %</td>
</tr>
<tr>
<td>22-22-11</td>
<td>standardisation</td>
<td>67.49 %</td>
</tr>
</tbody>
</table>

Figure 3-6. Classification results for 22-22-11 NN classifier with standardised data on 7 military (M1 - “Multi-function”, M2 - “Battlefield”, M3 - “Aircraft”, M4 - “Search”, M5 - “Air Defense”, M6 - “Weapon” and M7 - “Information”) and 4 civil classes (C1 - “Maritime”, C2 - “Airborne Navigation”, C3 - “Meteorological” and C4 - “Air Traffic Control”).

Although a straightforward comparison with radar classification studies reported by other authors might be misleading, due to the different data sets, model parameters and training methods used, the achieved results appeared to be strongly competitive when compared to the ones reported in (Anjaneyulu et al., 2009; Feng & Lin, 2014; Granger et al., 2001; Ibrahim et al., 2009; Lee, Choi, Cho, Rothwell, & Temme, 2014). Furthermore, additional improvement is expected, if further statistical pre-processing techniques, missing data handling routines, NN topologies or training algorithm parameters are investigated, as shown in the next two case studies.
3.4.2 Case Study 2 – Multiple Imputation and Feedforward Neural Networks

The second case study follows the same sequence of experiments and NN topologies, as introduced in the first study, however, this time an extended dataset, received after multiple imputation of the missing data values (as described in Section 3.3.1) is used.

For the purposes of the first experiment in this study, the categorical attributes of the input data are coded with consecutive integers. Two NN topologies are examined – 12-10-1 and 12-10-2, where the output parameter is coded as one binary neuron taking values 0 (“Civil”) and 1 (“Military”) for the first topology and 2 neurons, taking binary values 10 (“Civil”) and 01 (“Military”) for the second one.

The performance of each of the topologies is investigated, evaluated and compared using training with the original data (no pre-processing), and after normalisation and standardisation. The results are summarised in Table 3-15, showing up to 5% accuracy improvement for the case introducing imputation.

<table>
<thead>
<tr>
<th>Topology</th>
<th>Input data</th>
<th>% Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No imputation</td>
</tr>
<tr>
<td>12-10-1</td>
<td>no scaling</td>
<td>78.1</td>
</tr>
<tr>
<td></td>
<td>normalised</td>
<td>80.8</td>
</tr>
<tr>
<td></td>
<td>standardised</td>
<td>80.8</td>
</tr>
<tr>
<td>12-10-2</td>
<td>no scaling</td>
<td>80.1</td>
</tr>
<tr>
<td></td>
<td>normalised</td>
<td>81.6</td>
</tr>
<tr>
<td></td>
<td>standardised</td>
<td>82.1</td>
</tr>
</tbody>
</table>

Sample confusion matrices for a 12-10-2 NN classifier trained with normalised input data and a validation stop activated after 106 epochs are given in Figure
3-7. They demonstrate improved accuracy rates (especially for the “Military” class) when compared to the case studies using listwise deletion to cope with the incomplete data samples (Figure 3-4).

![Figure 3-7](image)

Figure 3-7. Classification results for imputed data case for 12-10-2 NN classifier with normalized input data and a validation stop after 106 epochs. The values in green specify the correctly classified samples for each class (0 for “Civil”, 1 for “Military”).

The second experiment in this study investigates two additional NN topologies – 22-22-1 and 22-22-2, where the output is again coded by one binary neuron (0 for “Civil” and 1 for “Military”) for the first topology and by two binary neurons for the second one (00 for “Civil” and 01 for “Military”). The NN performance for each of the topologies is investigated, evaluated and compared using the original, normalized and standardized data for both the cases – with and without imputed values. The performance results are summarised in Table 3-16, again showing improved NN performances for the cases with imputed data.
Table 3-16. Classification performance (over the testing set) for binary input coding and 22-22-N topologies with no data scaling, after normalisation and after standardisation. Comparison between NN training with data received after listwise deletion and after multiple imputation.

<table>
<thead>
<tr>
<th>Topology</th>
<th>Input data</th>
<th>% Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No imputation</td>
</tr>
<tr>
<td>22-22-1</td>
<td>no scaling</td>
<td>81.9</td>
</tr>
<tr>
<td></td>
<td>normalised</td>
<td>83.3</td>
</tr>
<tr>
<td></td>
<td>standardised</td>
<td>83.1</td>
</tr>
<tr>
<td></td>
<td>no scaling</td>
<td>81.8</td>
</tr>
<tr>
<td></td>
<td>normalised</td>
<td>83.9</td>
</tr>
<tr>
<td></td>
<td>standardised</td>
<td>84.3</td>
</tr>
</tbody>
</table>

The final experiment investigates a broader output space of 11 classes (4 “Civil” and 7 “Military”) and evaluates 22-22-11 NN classifiers with the original, normalised and standardised training data, and with dummy variable coded outputs. Summary of the obtained results when training on data subsets with and without imputation is presented in Table 3-17.

Table 3-17. Classification performance (over the testing set) for binary input coding and 22-22-11 topology with no data scaling, after normalisation and after standardisation. Comparison between NN training with data received after listwise deletion and after multiple imputation.

<table>
<thead>
<tr>
<th>Topology</th>
<th>Input data</th>
<th>% Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No imputation</td>
</tr>
<tr>
<td>22-22-11</td>
<td>no scaling</td>
<td>61.9</td>
</tr>
<tr>
<td></td>
<td>normalised</td>
<td>66.7</td>
</tr>
<tr>
<td></td>
<td>standardised</td>
<td>67.5</td>
</tr>
</tbody>
</table>

Sample confusion matrices for imputed 22-22-11 NN case trained with standardised input data and a validation stop activated after 98 epochs are presented in Figure 3-8. Although the results seem slightly inferior to the listwise deletion case (Figure 3-6), they give higher statistical confidence because of the increased number of samples.
Figure 3-8. Classification results for inputed data and 22-22-11 NN classifier with standardised data on 7 military (M1 - “Multi-function”, M2 - “Battlefield”, M3 - “Aircraft”, M4 - “Search”, M5 - “Air Defense”, M6 - “Weapon” and M7 - “Information”) and 4 civil classes (C1 - “Maritime”, C2 - “Airborne Navigation”, C3 - “Meteorological” and C4 - “Air Traffic Control”).

It can also be seen from Figure 3-8 that although the accuracy of the NN classifier is relatively the same (compared to the NN trained after listwise deletion (Figure 3-6)), the number of hits is largely increased and with a better distribution. This is especially evident for the ‘M7’ class, for which there were no hits in the case without imputation. The best accuracy is again achieved for the ‘M4’ and ‘C1’ classes, but the more important achievement as a result of the imputation is the uniform distribution of correctly classified samples. As illustrated in Figure 3-6, the class accuracy variance for the classification with no missing data is very high, from 0% to 87.9%, whereas in the case using imputed data (Figure 3-8), it is between 22.6% and 87.4%. In other words, while keeping the best accuracy almost the same, the minimum accuracy is improved by more than 22%. This should be attributed to the greater number of available training and testing samples as a result of the imputation, which increases the
statistical power of the dataset and subsequently improves the classification performance of the NN.

3.5 Conclusion

Reliable and real-time identification of radar signals is of crucial importance for timely threat detection, threat avoidance, general situation awareness and timely deployment of counter-measures. In this context, this chapter investigates the application of NN-based approaches for timely and trustworthy identification of radar types, associated with intercepted pulse trains, especially when large amount of missing data is present.

A number of experiments are designed, implemented, executed and evaluated for testing and validating the performance of the proposed intelligent systems for solving the investigated classification tasks. The different experiments study a variety of NN topologies, data transformation techniques and missing data handling approaches.

The simulations are divided in two broad case studies, each of which conducts several sub-experiments. In the first one, all the signals are pre-classified by experts into between 2 and 11 classes, depending on the experiment, and then a listwise deletion is used to clean the data from incomplete samples. As a result, very competitive classification accuracy of about 81%, 84% and 67% is received for the different recognition tasks.

In the second one, a study applying multiple imputation model-based approach for dealing with the large number of missing data (contained in the available radar signals data set) is investigated. The experiments conducted for the purposes of the first case study are repeated, but this time using the imputed data set for training of the classifiers. An improved accuracy of up to
87.3% is achieved. The results are compared and critically analysed, showing overall improved accuracy when the NN are trained on the larger subset with imputed values.

Although a straightforward comparison to radar classification studies, reported by other authors might be misleading, due to the different data sets, model parameters, data transformations, training and optimisation methods used, the achieved results are strongly competitive to the ones reported in (Anjaneyulu et al., 2009; Granger et al., 2001; Ibrahim et al., 2009; Lee et al., 2014; J. Li & Ying, 2014; Peipei & Hui, 2014).

Potential areas for further extension of this research include investigation of additional statistical transformation techniques, such as Principal Component Analysis (PCA), Non-Linear Principal Component Analysis (NLPCA), and Linear Discriminant Analysis, for decreasing the dimensionality of the problem and increasing the separability between the classes. In terms of classifiers, unsupervised learning techniques (such as self-organising maps (SOM)) can be considered, as well as varying other training parameters and exploring additional NN topologies. Finally, additional classes can be introduced, in order to achieve more specific classification of the intercepted radar data.

As a result of the research presented in this chapter, one journal ([3] in the journal publication list (Section 6.3)) and two peer-reviewed conference papers ([1, 2] in the conference publication list) have been accepted for publication and presentation. A book chapter ([1] in the books publication list) is also prepared and currently under review.
Chapter 4  Intelligent Optimisation of a Hydrotreating Process

This chapter addresses the problem of optimising a kinetic prediction model for simulating a major purifying process used in British Petroleum (BP) refineries, known as hydrotreating. The performance of the model is investigated over a set of 16 specially selected crude oil feeds. Global optimisation with genetic algorithms is applied over a number of model parameters and the simulation, testing and validation results show improved accuracy and efficiency. Several MATLAB functions and a graphical user interface (GUI) tool are designed and implemented during the course of the study. A comprehensive research report, alongside with the developed software, is delivered to an industrial partner (BP International Ltd) under a 6-month Knowledge Transfer Network (KTN) agreement to assist the company in further analysis, optimisation, testing and validation of similar prediction models (Appendix A3).

The rest of the chapter continues by providing relevant background knowledge and a review of related literature on the investigated problem (section 4.1), followed by a section on the current state of the studied prediction model (section 4.2) and a discussion on the adopted data acquisition and model gap identification steps (section 4.3). The undertaken optimisation approach is described in section 4.4, alongside with details on the development of a MATLAB-based GUI tool to assist the process. The final two sections present
the results from the optimisation (4.5) and conclude the chapter, providing ideas for further extension of this work (4.6).

4.1 Background Knowledge

Petroleum companies undergo continuous research in the way they process different crude oils, in order to meet the increased demand for cleaner fuels and petroleum products. This demand is dictated by more and more stringent environmental regulations that aim to reduce the pollution caused mainly by the sulphur and nitrogen components, present in these products (Alvarez, Ancheyta, & Muñoz, 2009; Y. Wang, Chen, & Larachi, 2013).

To meet the desired specifications, a number of technologies have been developed to refine and upgrade petroleum residues. Catalyst hydrotreating, as one of these technologies, is a key method adopted in many modern refineries (Alvarez et al., 2009; Ivanchina, Sharova, Anna, & Yakupova, 2014; Rana, Samano, Ancheyta, & Diaz, 2007). It has the capacity to increase the yield of distillates and to reduce the crude impurities, such as sulphur, nitrogen, or metals (J Ancheyta et al., 2002). Commonly, hydrotreating is carried out in reactors loaded with CoMo (Cobalt-Molybdenum) or NiMo (Nickel-Molybdenum) catalysts (Alvarez et al., 2009; Furimsky, 1998; Furimsky & Massoth, 1999).

The major issues with this technology are related to the catalyst deactivation, due to the presence of metals in the feeds and also the severity of the ongoing chemical reactions. The control of the reactor’s operation conditions (such as temperature, pressure, or space velocity) could have a drastic impact on the length of the catalyst life cycle (Kam, Al-Shamali, Juraidan, & Qabazard, 2005; Fabián S Mederos, Ancheyta, & Chen, 2009; Sie, 2001). This is of significant economic importance to the process, with respect to optimising the expenses
on catalyst deactivation, hydrogen consumption, heating, cooling, pumping, and compression in the reactors (Alvarez et al., 2009; Bej, 2002; Kam et al., 2005).

A good control of the hydrotreating process environment is beneficial for keeping the balance between the product quality parameters and the rate of catalyst deactivation (i.e. its cycle length). Therefore, the optimisation of the hydrotreating efficiency is of major interest for refineries (Alvarez et al., 2009).

To minimise the costs, time and effort required for experimental work, mathematical prediction models are used. However, developing a good model is not a trivial task, due to the complex physical and chemical transformations that the feeds undergo, as well as the mass-transfer phenomena and the catalyst deactivation mechanisms in the reaction system (Alvarez & Ancheyta, 2008).

Different approaches have been reported and most of them deal with trickle-bed reactor models, applied to the hydrotreating of middle distillates (Avraam & Vasalos, 2003; Bhaskar, Valavarasu, Sairam, Balaraman, & Balu, 2004; Chowdhury, Pedernera, & Reimert, 2002; Ivanchina et al., 2014; Murali, Voolapalli, Ravichander, Gokak, & Choudary, 2007) and vacuum gas oils (H. Kumar & Froment, 2007; Fabian S Mederos & Ancheyta, 2007; Rodriguez & Ancheyta, 2004). Only a few models (developed by major petroleum companies) have been developed for heavy feeds. A more comprehensive summary of available models can be found in (Alvarez & Ancheyta, 2008; Jorge Ancheyta, Sanchez, & Rodriguez, 2005; Y. Wang et al., 2013).

In order to safely monitor and optimise the operation of its hydrotreating units, BP has developed a computer simulation model. It provides the ability to predict the effects that different feeds have on an apparent catalyst activity. The prediction model has played an essential role in monitoring, optimising, revamping and designing new hydrotreating units across the company.

However, as the environmental regulations and market demands for cleaner fuel become more stringent, BP faces new challenges in operating these units.
The observation and critical analysis of many short catalyst life cycles of company’s diesel hydrotreaters, while producing ultra-low sulphur diesel (ULSD), have shown that variations in the feed quality can have a significant impact. For example, running distillates from troublesome crudes can shorten the catalyst life by 30%, causing significant financial losses.

To address these problems, BP has commissioned a comprehensive pilot plant study to investigate and simulate the impact of different feedstocks under the ULSD operation conditions.

The next subchapters of this study focus on investigation, identification and analysis of the gaps between the measured pilot plant data and the predictions made by the current BP’s hydroprocessing mathematical model and propose an approach to optimise and improve its accuracy and efficiency.

### 4.2 Investigated Approach

#### 4.2.1 Status Quo

The investigated hydroprocessing model is based on the first principle of thermodynamics and is implemented in FORTRAN as a dynamic-link library (dll). It takes as inputs more than 250 parameters and produces more than 60 outputs. Some of those inputs are specific to the individual crude feeds, while others are related to the kinetics of the chemical reactions and the technologies used in a given refinery (i.e. some of them are constant for a specific refinery).

The available interface for interaction with the model is written in VBA (Visual Basic for Applications) and uses Microsoft Excel as an environment for receiving the model inputs and presenting the predicted results.
For the purposes of this study and for maintaining confidentiality, the hydrotreating model will be further discussed as a ‘black box’, without giving too much detail about the way it works, but focussing our attention on how its accuracy can be improved.

Although this model has played an important role in monitoring, optimising and designing new hydrotreating units, its performance is weakened by some gaps in its prediction capabilities that need to be addressed, after detailed investigation and analysis.

### 4.2.2 Approach Overview

As many repetitive tasks are to be performed, including tests on different hydrotreating model versions using available plant data sets, a way of automating the model execution and data manipulation is needed. MATLAB® has been chosen as the main working environment for this investigation because of its advanced analysis, optimisation and visualisation tools as well as its rich high-level programming language.

In this context, the first task is to investigate how to link the existing hydrotreating model to MATLAB, in order to import and export data, and how to set up various model parameters and run the model with some basic options within MATLAB.

Afterwards, gap identification and analysis are performed on the investigated hydrotreating model. Comparison between the predicted and desired outputs are made. The available data is considered feed by feed, reactor by reactor and as a whole.

Next, several more advanced manipulation functions are implemented (such as running hydrotreating plans, feed data extraction, reactor data extraction, results visualisation, exportation, etc.), in order to facilitate the model
optimisation process. Most of them are included in a graphical user interface (GUI) tool for enhanced interaction, analysis and optimisation.

Finally, model optimisation incorporating genetic algorithms is performed for improving the model prediction capabilities.

4.3 Gap Analysis

The characteristics of the products delivered by refineries are required to meet specific regulation criteria (Alvarez et al., 2009; Y. Wang et al., 2013). Although producing low sulphur and nitrogen species is highly desirable from an environmental point of view, it has to trade off with very high manufacturing costs. Therefore, a good hydroprocessing simulation model is of great importance for the accurate monitoring and control of BP’s hydroprocessing units. As the available kinetic model experiences some difficulty in simulating the impact of different feedstocks, especially under ULSD conditions, this section focuses on identifying gaps between the measured data and the predictions made by the model.

4.3.1 Acquired Data

The data used in this research is acquired by an industrial partner of this project (BP International Ltd) by running a pilot plant study. It is conducted concurrently in 4 small (pilot plant) reactors under three different conditions (temperature of the reactor beds, pressure, etc.) over a period of about 2 months. Sixteen diverse feeds, covering BP sources from around the globe are examined, including straight run (SR) diesel feeds, light cycle gas oil (LCO), light coker gas oil (LCGO) and distillates from residue hydrocracking units (RHU). The different feed processions are split between the reactors (four in
each reactor) and a reference feed is run before loading the next crude, in order to record the catalyst activity decrease.

Observations for each condition are made and a number of relevant parameters are recorded. In this way, a data set containing information for 64 different experimental conditions is generated (its entries will be further referred to as condition points). Each condition point consists of a vector of refinery measured data (such as reactor bed temperature, pressure, etc.), taken at a specific moment of time.

4.3.2 Gap Analysis Approach

The acquired data is used to extract 64 vectors, describing each of the condition points that are to be fed as inputs to the model. Most of the input vector elements are direct result from the measurements, while others are derived after interpolation of the three condition points recorded for each feed. The MATLAB’s polynomial interpolation function polyfit (http://www.mathworks.co.uk/help/matlab/ref/polyfit.html) is used to estimate those values.

After deriving the input vectors for the model, it is run and the predicted results are recorded. The model’s performance is analysed with respect to the amount of sulphur and nitrogen in the end product, as this has been the area of significant interest. The work is focused mainly on those two parameters, because of their importance to the product quality.

The analysis is performed using MATLAB and its Global Optimisation and Statistics Toolboxes. Graphics showing the relationship between the measured and the predicted results are created, critically analysed and compared subsequently.
4.3.3 Sulphur Gap Analysis

The model's sulphur prediction performance is shown in Figure 4-1. The x-axis represents the measured product sulphur values in parts per million (ppm) and the y-axis – the predicted sulphur values in ppm. The diagonal line shows where the condition points should lie if the sulphur values, predicted by the model, coincide with the measured ones. Each condition point in the figure represents a single entry in the database and no additional information is given about it, in order to preserve the confidentiality of the data and because that information is not within the main focus of the investigation.

Figure 4-1. Sulphur prediction performance of the hydrotreating model. Ideally, the condition points should lie on the diagonal line (i.e. the measured values to coincide with the predicted ones). Gaps of more than 50% of the measured values are observed.

The results show that, in general, the model underestimates the product sulphur rates. The mean gap between the predicted and the measured sulphur within the product is 8.78 ppm, whereas the peak gaps could reach up to 50 ppm (or more than 50% of the measured values, e.g., the last three conditions in Figure 4-1) for some of the troublesome feeds.
4.3.4 Nitrogen Gap Analysis

The nitrogen prediction performance results are shown in Figure 4-2. Analogically to the sulphur analysis case, the measured total product nitrogen is given on the x-axis and the predicted one – on the y-axis. The diagonal line represents a reference line, where the condition points should lie in the ideal case.

![Nitrogen prediction performance of the hydrotreating model](image)

**Figure 4-2.** Nitrogen prediction performance of the hydrotreating model. Ideally, the condition points should lie on the diagonal line (i.e. the measured values to coincide with the predicted ones). Gaps of up to 60 ppm from the measured values are observed.

The analysis of the results shows that the model has serious problems in the prediction of the total nitrogen within the product. A mean gap of 5.03 ppm is calculated, where the maximal discrepancy could reach up to 60 ppm (the conditions in the far right in Figure 4-2) for some of the condition points. However, analysis also shows that most of the condition points lie on a line, indicating good opportunities for optimisation.
4.4  Model Optimisation

The optimisation is performed by tuning some of the kinetic model setup parameters, so that it could better predict the product sulphur and nitrogen values. As the logic incorporated in the model is complex and the functional relations are not necessarily continuous, a genetic algorithms optimisation approach is chosen, when searching for a global solution.

A group of 20 model setup parameters is chosen or evaluation. The selection of these parameters is driven by their significance and influence on the model’s prediction performance, as well as on the chemical reaction expert knowledge incorporated in the model. The investigated parameters are related to the pre-exponential factors and activation energies for some important chemical reactions embedded in the model.

The process of optimisation is performed in the MATLAB® environment using its Global Optimisation Toolbox functionality and is discussed in more detail in the next subsections.

4.4.1  Linking the Model to MATLAB®

Two approaches are considered for linking the hydrotreating model to MATLAB®. First, the possibility of using the already created Excel model interface is investigated. The process of importing data, exporting data and execution of an Excel macro in MATLAB® is a relatively simple and easy task. However, the main disadvantage of this approach is the added loss of performance. The model’s execution time is of crucial importance, especially when hundreds of thousands of test runs are to be evaluated. Therefore, another approach is necessary and functionality for calling the hydroprocessing model through its FORTRAN dll is developed. This is not a
straight forward process, as a direct calling of FORTRAN dlls is not currently supported by MATLAB®. To resolve this issue, a header file in C++ style is defined for the dll and the C++ import procedure is followed. As a result of this approach, the execution time needed for evaluation of a single test condition is reduced by five times, compared to the initial approach.

4.4.2 GUI Tool for Model Analysis and Optimisation

For speeding up the model analysis, optimisation, testing and validation, a MATLAB®-based graphical user interface tool is designed, developed and implemented. It includes functionality for: defining, saving and loading feed and reactor data; for defining, saving, loading and running of refinery reactor work plans; and for running optimisations and testing and validation of the optimised models. The main screen of the developed GUI tool is shown in Figure 4-3.

![Figure 4-3. Main screen of the GUI tool, developed in MATLAB® for analysis, optimisation, testing and validation of the investigated prediction model.](image)
In addition, a module for generation of comparison plots and visualisation of the prediction performances is implemented to further enhance the investigation, exploration and analysis of the studied model.

The implemented GUI tool is used to conduct most of the experiments and to generate the corresponding optimisation result plots, presented in this chapter.

### 4.4.3 Optimisation

A set of 20 model parameters is selected or optimisation in this chapter based on chemical engineers advice and expertise. They are divided into three broad groups based on their relation to the prediction parameters of interest – 8 related to the product nitrogen, 10 related to the product sulphur and 2 related to the product nitrogen and sulphur at the same time.

Global optimisation, employing genetic algorithms is applied. The mean squared error (MSE) between the measured data \((M)\) and the one predicted by the model \((P)\), over \(n\) condition points is chosen as a fitness function for the optimisation \((4.1)\), in order to emphasize the reduction of the largest prediction gaps (Georgieva & Jordanov, 2010; Marsland, 2011).

\[
F = \frac{1}{n} \sum_{i=1}^{n} (P_i - M_i)^2
\]  

\((4.1)\)

Constraints for each of the optimised setup parameters are defined, so that they could take chemically reasonable values. The optimisation process is divided into 5 steps. First, the model is optimised for 8 setup parameters, related to the nitrogen predictions. This is done in two steps – in groups of four, based on the nature of the parameters and chemical engineering expertise. The nitrogen’s performance optimisation is conducted prior to sulphur, because of the inhibition effects that nitrogen could have on the product sulphur values. Next, a two-step optimisation is performed for the sulphur related parameters – in
groups of five. Again, the grouping of the parameters is based on their chemical nature. Finally, an optimisation of two, related to both sulphur and nitrogen parameters is conducted.

Chromosomes with two, four or five genes, representing the groups of sulphur and nitrogen parameters are used. Starting with a population of 20 chromosomes, different selection techniques (stochastic uniform, reminder, roulette, and tournament), and reproductive operators - cross-over (scattered, single point, two points) and mutation (Gaussian, constraint dependent) are used for the genetic algorithms runs. The stopping condition is set to reach 100 generations or a zero error (whichever occurs first).

At each stage, the setup parameter values that are to be accepted for the optimised model are decided by refinery chemical experts, based on the model’s prediction performance and the chemical consistency of the combination of the parameter values. This is necessary, because solutions with an insignificant difference in the value of the fitness function sometimes result in a chemically more sound combination of parameters.

### 4.5 Optimisation Results

The performance of the optimised model is compared with the initial model. Its prediction capabilities are examined feed by feed, reactor by reactor and as a whole. An example of the improved results from the optimised model for a randomly chosen feed is shown in Figure 4-4. The blue line represents the measured sulphur data and the red and green lines – the predictions made by the initial and the optimised models, respectively. The reactor’s temperature (as one of the important inputs) is shown on the $x$-axis. The labels on the points denote the condition indices in the data set. It can be seen that the prediction gaps have been reduced for the optimised model and its predictions follow
better the trend of the measurements. As a result, the mean performance gap for the feed is almost halved - from 6.38 ppm to 3.68 ppm.

![Graph showing product sulphur optimisation results](image)

**Figure 4-4.** Product sulphur optimisation results for a sample feed. The mean prediction gap is almost halved.

The comparison results of the initial and the optimised model in predicting the product sulphur for all the feeds are shown in Figure 4-5. The predictions made by the optimised model are given with green markers and as it can be seen they have shifted closer to the desired diagonal line. As a result, the mean difference for the sulphur predictions is reduced from 8.78 ppm for the initial model, to 4.60 ppm. Also, the model reliability is improved and the peak sulphur prediction gaps are closed from 50 ppm to up to 20 ppm.
Figure 4-5. Product sulphur optimisation results for all studied feeds. The mean prediction gap is almost halved.

Analogically, Figure 4-6 presents the model comparison results for the same randomly selected feed given in Figure 4-4, but this time for the product nitrogen predictions. It can be seen that the optimised model not only gives results closer to the measured values, but also follows better the trend of the product nitrogen change when varying the reactor inlet temperature. The improvement is more explicit for the lower temperature range, where the gap is closed from about 12 to 5 ppm. As a result, the mean difference is reduced from 4.96 ppm to 2.87 ppm.

Figure 4-6. Product nitrogen optimisation results for a sample feed. The mean prediction gap is almost halved.
The combined optimisation results for the product nitrogen predictions for all the feeds are given in Figure 4-7. As it can be seen, the model predictions have shifted closer to the reference line. As a result of the optimisation, the mean discrepancy of the nitrogen predictions is reduced from 5.03 ppm to 2.66 and the peak gaps are closed from 60 ppm to up to 10 ppm.

![Figure 4-7. Product nitrogen optimisation results for all studied feeds. The mean prediction gap is almost halved.](image)

The overall analysis shows that in general, the considerable improvement of the model prediction capabilities can be attributed to the proposed GA-based optimisation approach. However, it has to be also noted that a good room for improvement had been present in the initial model due to the model been developed by experts only to the best of their knowledge and expertise and without almost no statistical or computational intelligence optimisation on the model parameters.

Although a straight forward comparison with other hydrotreating mathematical models may be misleading (due to the different nature of the underlying computational apparatus, model input parameters, setup parameters, error measurements and test data (size and feed’s diversity)), it has to be noted that the optimised model prediction results are in good agreement
or even superior to those of models reported by other authors within similar works.

For example, in (Marafi, Maruyama, Stanislaus, & Kam, 2008), the authors propose a catalyst evaluation methodology that allows the kinetics of apparent reactions and catalyst life to be predicted and also report very satisfactory results from both pilot plant and commercial plant tests. Again, good agreement of the predicted values in the studied range, when compared to the experimental data, is achieved by the model, presented in (Alvarez & Ancheyta, 2008). In it, the reactor setup parameters are optimised following the Levenberg-Marquardt’s algorithm for minimising the sum of the squares of the residuals between the experimental data and the model predictions. The same algorithm for obtaining the optimal set of kinetic parameters is employed with similar success in (Alvarez et al., 2009). In (Alvarez, Ancheyta, Centeno, & Marroquín, 2011) a hydroprocessing model taking into account more reactor configuration properties is investigated and the reported predictions’ accuracy falls into a ±10% interval. Good modelling results are also achieved in (Kam et al., 2005; Sánchez, Rodríguez, & Ancheyta, 2005; Toulhoat, Hudebine, Raybaud, Guillaume, & Kressmann, 2005; Umana, Shoaib, Zhang, & Smith, 2014).

### 4.6 Conclusion

A model optimisation approach for improving the accuracy and efficiency of an existing hydroprocessing model is studied in this chapter. The prediction gaps of the model are investigated and a set of its setup parameters are chosen for consideration. Global optimisation for the selected parameters is performed using genetic algorithms on a specially composed diverse feedstock data set.
As a result, the model performance becomes more reliable as a whole and the peak differences between the predicted and the experimentally measured values are reduced up to 20 ppm for the sulphur and up to 10 ppm for the nitrogen. This will result in providing better opportunities for planning the operation of the hydrotreating units of the partnering company and improving the economics of the process by reducing the feed processing time, hydrogen consumption, catalyst deactivation rate, etc.

As a continuation of the work, further analysis and evaluation of the optimised model can be conducted for a broader pilot and commercial plant data. In addition, the same or very similar approach can be considered for improving other prediction models, especially if they are only relying on expert knowledge.

As a result of the research presented in this chapter, one peer-reviewed conference paper ([3] in the conference publication list (Section 6.3)) has been accepted for publication and presentation. A Case Study has also been published for disseminating the results (Appendix A3).
Chapter 5  Intelligent Parametric Patch Perturbation for Aerodynamic Surface Optimisation

The aerodynamic design industry needs a tool that can help their engineers to fine-tune the aerodynamic parameters of a shape they have been working on. Typical shapes for optimisation include wings, winglets, pylons, fuselages, etc., where the objective will normally be to maximise the configuration’s lift force or to minimise its drag.

The industry demands such tools to be efficient enough, in order to assist the designers in an interactive shape creation and modification process. The suggested shape modifications are expected to comply with given design constraints and to be intuitive and easy to track and analyse.

Taking into account that the objective functions are normally defined in a complex closed-code and computationally very expensive commercial products (Forrester & Keane, 2009) (it takes a few hours for a single run to be executed on supercomputers), the main motivation of this research is to investigate, implement, test and validate a parametric approach for automated aerodynamic optimisation of geometry configurations, that can converge fast enough to an optimum (times faster than existing approaches), for adoption during a manual shape design process.
The method, introduced in this chapter, explores practical heuristics in which the aerodynamic pressure is directly linked with the geometry’s curvature and tangent (depending on whether the flow is subsonic or supersonic), to derive sets of basis functions (for spanning the design space) that can significantly facilitate the work of the optimiser in finding better shape configurations. In particular, a surrogate objective function (related to the streamwise curvature of the perturbed area) is used for calculating the basis functions to be as close to orthonormal as possible with respect to an objective function based on an expensive computational fluid dynamics (CFD) simulation. Several large simulations are run on supercomputers through an industrial partner (AIRBUS Operations Ltd, Case Study is presented in Appendix A4) and the initial results indicate very promising improvement in the convergence rate and highlight opportunities for efficient automatic shape optimisation applications.

The rest of the chapter continues by a literature review (section 5.1) on the limited research available on the investigated topic. Next, the key ideas behind the implemented solution (section 5.2) are presented, followed by a subchapter (5.3) discussing in detail the suggested approach. The derivation of the used basis functions and the implemented software solutions are presented in section 5.4 and section 5.5, respectively. The initial results from running large simulations with real aircraft design surfaces are shown in 5.6. Finally, section 5.7 summarises the chapter and gives ideas for further extension of this work.

5.1 Background Knowledge

In a highly competitive aircraft industry, it is no longer sufficient to design an aircraft product whose performance is just satisfactory. It is essential to build products that can compete and outperform the ones currently available on the market. To succeed in this, the modern aircraft design process focuses and relies more and more on the development and evaluation of new theoretical
concepts and approaches that are capable of providing the desired results in a timely and economic manner (Forrester & Keane, 2009; Iuliano & Quagliarella, 2013b).

Aerodynamic shape optimisation tools reduce the design time, improve the design quality and free engineers and designers for creative work by taking over some tedious operations. However, this process normally involves a huge number of repetitive simulations in a search for an optimal design candidate, where each simulation requires execution of a very computationally intensive evaluation procedure (that involves solving complex systems of partial differential equations (PDE) until convergence) (Iuliano & Quagliarella, 2013a). This is particularly true when the aerodynamic optimisation is based on computational fluid dynamics (CFD) methods.

In order to reduce the computational cost, the scientific community is focussing on alternative methods, such as surrogate and reduced order modelling that are aimed to provide an approximation of a selected objective function over a design space for a significantly reduced cost (Iuliano & Quagliarella, 2013b).

For example, in (Giunta, Watson, & Koehler, 1998; Goel, Hafkta, & Shyy, 2009), methods exploiting polynomial surfaces are investigated for surrogate-based optimisation. Another popular approach is to use a weighted sum of simple functions (Radial Basis Functions (RBFs)) in an attempt to emulate complicated design landscapes (Boulkeraa, Ghenaiet, Mendez, & Mohammadi, 2013; Keane, 2004; Rendall & Allen, 2009).

Several applications of Kriging (a mathematical method based on statistics, which is used to interpolate the value of a random field at an unobserved location, taking into account observations at nearby positions) have also been investigated in the literature and have confirmed its ability to approximate multi-dimensional, highly-nonlinear functions (Forrester & Keane, 2009; Forrester, Sóbester, & Keane, 2007; C. Li, Brezillon, & Görtz, 2011). Several Kriging modifications are available, including: simple Kriging (assumes a
known constant trend (Forrester et al., 2007; J. Toal, Bressloff, & Keane, 2008; Liao, Liu, & Long, 2011); ordinary Kriging (supposes an unknown, but predicted constant trend (Han, Görtz, & Hain, 2010)); and universal Kriging (presumes low-order polynomial as trend (Forrester & Keane, 2009)). A review of additional surrogate-based optimisation techniques can be found in (Forrester & Keane, 2009) and implementations of reduced order models (mainly using proper orthogonal decomposition (POD)) are discussed in detail in (Andoh, Shimoyama, Jeong, & Obayashi, 2012; Iliescu & Wang, 2014; Iuliano & Quagliarella, 2013a, 2013b).

A typical objective of an aerodynamic shape optimisation problem is the drag minimisation of the investigated shape by modifying its surface. As the drag is an integral of force (pressure and shear) over the geometry, the majority of aerodynamic optimisation researchers use design parameters that are defined directly in the geometry shape. Popular examples are Hicks-Henne (Hicks & Henne, 1978) bump functions (analytical shape functions), PARSEC (PARametric SECtion) (Derksen & Rogalsky, 2010; Sobieczky, 1999), Class-shape function transformation (CST) method (Kulfan, 2008), B-spline (Nemec & Zingg, 2002) and surface mesh points (Jameson, 2003; Leoviriyakit, 2005).

Selecting a suitable parameterisation requires a compromise between cost (using as fewer parameters as possible) and the flexibility to capture a true optimum shape (requiring a very large number of design parameters).

In this work, assumptions suggested by analytical aerodynamic theories (Sinnott & Osborne, 1961; Spreiter & Alksne, 1958) that the aerodynamic pressure is directly linked to the geometry curvature and tangent (depending whether the flow is subsonic or supersonic) are investigated. In this context, in order to have greater impact on the pressure changes, an idea to use curvature to drive the design, rather than the geometry itself, is adopted. Therefore, the research is focused on aerodynamic optimisation that is driven by the curvature...
space, where the geometry changes will simply be the result of curvature changes.

5.2 Key Ideas

Normally, the commonly applied approaches for assessing the aerodynamic properties of an investigated geometry include performing complex and computationally very expensive evaluation procedures, implemented in costly commercial CFD products (Iuliano & Quagliarella, 2013a).

The objective functions defined in these products usually output values that indicate the goodness of the investigated shape in terms of lift, drag, stress, etc. These values are then used by experts or optimisation algorithms to decide what changes are to be introduced in the designed geometry, in order to improve its aerodynamic properties.

However, as the evaluation CFD procedure is very often embedded in commercial products with no direct access to the program code or to the analytical form of the underlying equations, it is hard to implement efficient automatic optimisation techniques that can run fast enough, in order to be adopted in a manual shape design process (Forrester & Keane, 2009; Iuliano & Quagliarella, 2013a, 2013b).

In an optimisation context, the space of the possible geometric configurations is to be spanned by a set of design variables. These variables can be used to control the changes to be made in an investigated configuration and by modifying them, the optimiser creates the next configuration that is to be analysed. For some very basic geometries (regions of surfaces, for example), the nodes of the mesh grid that hold the shape can be chosen as design variables, however, in the common case their number is far too large to be used as such.
The key idea behind the investigated approach is to derive a *surrogate* objective function (in an analytical form) that is likely to be closely correlated to the *real* commercial CFD objective function (used during the optimisation) and that can be efficiently computed for a given geometric configuration.

This *surrogate* function is then used to facilitate finding a set of *basis* functions for spanning the design space that would make the optimisation of the *real* objective function *well-conditioned* (i.e., small changes in its input parameters to result in small output changes), so that the optimiser can converge in fewer iterations of the computationally expensive evaluation process.

In mathematical context, a good *surrogate* function would be one for which a basis chosen to be *orthonormal* with respect to it (*orthonormality* will be discussed in detail in the next section) happens to be also *well-conditioned* for the *real* objective function. Then, if the *surrogate* is good in the above sense, this could result in a really efficient optimisation (Fletcher, 2013).

Although there is a potentially unlimited number of possible *surrogate* functions that can be exploited, available insights of the physics behind the evaluation procedure, as well as some observed empirical correlations for the *real* objective function, can help in selecting good *surrogates*. For example, the fact that the pressure variations on a physical surface are closely related to its curvature (Sinnott & Osborne, 1961; Spreiter & Alksne, 1958), suggests that functions of curvature could be good choice for *surrogates*.

Finally, once a suitable *surrogate* function is chosen, a basis for spanning the design space by the optimiser can be derived in algebraic form, such that it is *orthonormal* with respect to the chosen *surrogate*. 
5.3 Investigated Approach

5.3.1 Scenario

The investigation starts from an initial geometry (wing, winglet, fuselage, etc.) that an aerodynamic design engineer generates using their best practice knowledge. Then, analysis is performed to determine areas in it, in which it would be most beneficial to introduce shape changes. Next, a geometric patch or a set of patches is applied on the selected area (Figure 5-1) and appropriate parameterisation is conducted. Finally, optimisation of the aerodynamic parameters of the investigated geometry is performed by introducing modifications to the shape of the selected patch surfaces by following the proposed approach. The way these modifications are introduced aims to facilitate the rapid convergence of the optimiser, which is the key objective of this research.

Figure 5-1. Rectangular patch applied over a wing shape area, selected for optimisation. The arrows show the directions of the local surface normals.
5.3.2 Geometry Displacement

The analysis starts from an initial geometry that is generated by aerodynamic design engineers, using external computer-aided design (CAD) software (such as CATIA\(^4\)), and is available as a set of points defined in a mesh grid. For simplicity in illustrating the approach, it is assumed that the selected geometry patches are of rectangular shapes, aligned to the local airflow. In practice, the latter is a reasonable assumption, however, using more complex patch forms is also possible, subject to an appropriate parameterisation (e.g., by applying transfinite interpolation, such as Coons patch (Farin & Hansford, 1999)). Also, in order to simplify the mathematical operations, the patch is defined in a parameter domain \((s, t)\), where \(s, t \in [0, 1]\).

Let the initial geometry be given by \(P_0(S)\), where \(P\) is a point in the real space \((x, y, z)\), and \(S\) is a position in a rectangular parameter space \((s, t)\). Then each new geometry \(P(S)\), introduced during the optimisation process, can be expressed by adding a set of displacements to the initial surface \(P_0(S)\), where each displacement \(v_iF_i(S)\) is received as a product of the \(i^{th}\) basis function \(F_i(S)\) from a set of \(M\) basis functions and a weighting scalar \(v_i\) that defines its amplitude.

\[
P(S) = P_0(S) + \sum_{i=1}^{M} v_i F_i(S).
\]  (5.1)

In equation (5.1), the element \(F_i(S)\) is a vector, where its magnitude \(f_i(S)\) is a scalar function of \(S\) and its direction at a given point \(S\) is taken to be that of the surface normal \(N\) of \(P_0\) at that point. Therefore:

\[
F_i(S) = f_i(S) N(P_0(S)).
\]  (5.2)

\(^4\)http://www.3ds.com/products-services/catia/
Choosing these scalar functions is of main importance for addressing the investigated problem.

As the initial surface $P_0$ will not change during the optimisation, the local surface normal $N$ will not vary significantly at a given point and therefore it can be ignored during the analysis.

### 5.3.3 Objective Function

The used commercial CFD objective function (DLR, 2012) results in a single real number, which changes with the values of the design variables. The relationships within the function are subject of large computations and no analytical form is available (the code is closed-source), although some insights of the relationships are attainable by derivative approximation methods, such as Finite Difference (Larson, 2012), for example.

The understanding of the commercial CFD objective function $c$ is that, near the optimum, it is a quadratic function of the design variables (DLR, 2012; Sinnott & Osborne, 1961; Spreiter & Alksne, 1958). In the investigated approach, these variables will be the weights of the displacements generated by each basis function (i.e., the coefficients $v_i$ in equation (5.2)). Also, as the design variables have an effect on the shape of the generated displacement throughout the whole surface of the parameter space $S$ and, at the same time the CFD function produces just a single scalar. It is implied that the relationship in that function involves a kind of weighted integration of some properties $\Phi(P(S))$ (that might include derivatives, as well) over $S$.

$$ c = \int_S \Phi(P(S)) dS. \quad (5.3) $$
5.3.4 Mathematical Definitions

Orthogonality

By definition, two functions are **orthogonal** with respect to an objective function $c$, if the first derivative of $c$ with respect to the amplitude $v_1$ of the first function (equation (5.1)) is not altered by small variations in the amplitude $v_2$ of the second one:

$$\frac{d^2c}{dv_1dv_2} = 0. \quad (5.4)$$

In other words, if the inner product of two functions is defined by $\int_a^b \phi_1(x)\phi_2(x)\,dx$, a set $\{\phi_1(x), \phi_2(x), \ldots, \phi_N(x)\}$ is an orthogonal set of functions in the interval $[a, b]$ if the inner product of any two of the functions is equal to zero:

$$\int_a^b \phi_n(x)\phi_m(x)\,dx = 0; \ n, m = 1, \ldots, N; \ \forall n \neq m. \quad (5.5)$$

Normality

A set $\{\phi_1(x), \phi_2(x), \ldots, \phi_N(x)\}$ of functions is a **normal**, if for each function $\phi_i(x)$:

$$\int_a^b (\phi_i(x))^2\,dx = 1, \ i = 1, \ldots, N. \quad (5.6)$$

Orthogonormality

A basis is **orthonormal** if every pair of functions in it is both **orthogonal** and **normal**. In this approach, if the basis functions, derived to be **orthonormal** with respect to the **surrogate** objective function, happen to be also **orthonormal** with respect to the **real** CFD objective function, then the optimiser should be able to reach the optimum within just a single line-search (Fletcher, 2013). Although
this is not to be expected in practice, because of the several assumptions made when deriving the surrogate objective function, the optimiser should be able to converge much faster than if no orthonormal basis is approximated for spanning the search space.

5.3.5 Streamwise Curvature

According to (DLR, 2012; Sinnott & Osborne, 1961; Spreiter & Alksne, 1958), it is well grounded to assume that the real objective function is strongly coupled with the streamwise curvatures of the displaced surface (at least in the case of subsonic speeds) and that is a heuristic which is going to be exploited in this approach. In other words, a distinction between the two parametric variables, which span the surface $P(S)$, is drawn. The variable $s$ selects which streamwise section is being chosen and $t$ varies monotonically along that section. If all changes of the tangent direction $s$ remain small, the streamwise curvature can be approximated as follows:

$$\frac{d^2P(S)}{dt^2} \cdot \left| \frac{dP(S)}{dt} \right|.$$

Additionally, if the domain is selected to be geometrically close to rectangular one, then the denominator in equation (5.7) will be close to constant and can be excluded during the analysis.

5.3.6 Tensor Product of Basis Functions

In terms of simplifying the calculations needed, the patch surface selected for optimisation is defined in a rectangular domain with axes $s,t \in [0,1]$. A
straightforward approach to allow variation in both directions of the domain is to give the list of basis functions spanning it a tensor product structure (Lehmann & Reif, 2012). In this way, the functions $f_i(S)$ from equation (5.2) will be selected by two indices: the first one related to the section direction ($s$); and the second one – to the position in the streamwise direction ($t$).

For $M$ basis functions defined in the cross-stream $\varphi_i$ and $N$ basis functions in streamwise $\psi_j$ direction, the resultant tensor product can be expressed as:

$$f_{ij}(S) = \varphi_i(s)\psi_j(t), \quad i = 1, \ldots, M, \quad j = 1, \ldots, N.$$  \hspace{1cm} (5.8)

### 5.4 Derivation of Basis Functions

#### 5.4.1 Functions with Respect to s and t Directions

As there are practical grounds for distinguishing between the cross-stream ($s$) and streamwise ($t$) directions in the design domain, different sets are needed for the $\varphi_i(s)$ and $\psi_j(t)$ functions. In order to follow the assumptions made for the CFD objective function (Section 5.3), the $\psi_j(t)$ basis functions should be orthonormal with respect to second derivatives of the objective function and the $\varphi_i(s)$ functions – with respect to its values. Then, in mathematical terms, the tensor product of the functions will result in being orthonormal with respect to the second derivatives in the streamwise direction.
5.4.2 Continuity of Position, Slope and Curvature

An important requirement of the investigated approach is to ensure that the perturbation of the geometry is continuous in position, slope and curvature in the interior of the selected optimisation area and at its edges. This would account for that patches, found to improve the aerodynamic parameters of a specific shape configuration, can be smoothly applied on already existing physical surfaces.

In the patch interior, these conditions can be achieved by deriving basis functions that are continuous and twice differentiable. At the edges they should have their value, first and second derivatives equal to zero, to guarantee smooth transition at the patch boundaries.

The first part of the conditions is automatically met by polynomials, which are continuous for all derivatives. The boundary requirements are attained by ensuring that all basis functions have triple roots at each of the edges. Then, for $M$ basis functions used in cross-stream $\phi_i$ and $N$ basis functions in streamwise $\psi_j$ directions, the relevant polynomial equations can be expressed as:

\[
\phi_i(s) = s^3(1-s)^3 \left( \sum_{k=0}^{i} a_k s^k \right), \quad i = 1, \ldots, M, \tag{5.9}
\]

\[
\psi_j(t) = t^3(1-t)^3 \left( \sum_{l=0}^{j} b_l t^l \right), \quad j = 1, \ldots, N. \tag{5.10}
\]

Subsequently, the derivation of good basis functions will depend on calculating the values of the coefficients $a_k$ and $b_l$ that will result in the desired orthonormality with respect to the objective function.
5.4.3 Odd and Even Functions

As any *symmetric* function is automatically *orthogonal* to all *asymmetric* functions (and vice versa) (Larson, 2012), a simple transformation of the functions’ domain can further reduce significantly the amount of the necessary calculations. If, instead of using the interval \([0, 1]\), the functions are defined in a new domain \(u, v \in [-1, 1]\), this would make it easy to distinguish between *symmetric* and *asymmetric* functions, where the symmetric functions will be of even degree and the asymmetric of odd. This can be done by the following substitutions:

\[ u \equiv 2s - 1, \quad (5.11) \]
\[ v \equiv 2t - 1, \quad (5.12) \]

so that the functions in terms of \(u\) and \(v\) (instead of \(s\) and \(t\)) are centred in the origin.

5.4.4 Cross-stream functions

Considering equations (5.5) and (5.6), the orthogonality and normality conditions in the cross-stream direction will take the form:

\[ \int_{u} \varphi_{i}(u)\varphi_{j}(u) du = 0, \quad \forall i, \forall j > i, \quad (5.13) \]
\[ \int_{u} (\varphi_{i}(u))^{2} du = 1, \quad \forall i. \quad (5.14) \]
5.4.5 Streamwise Functions

Following the assumption outlined in Section 5.3 that the objective function is closely correlated to the streamwise curvature and again taking into account equations (5.5) and (5.6), the orthogonalisation and normalisation conditions in the streamwise direction can be expressed as:

\[
\int v \left( \frac{d^2 \psi_i(v)}{dv^2} \right) dv = 0, \quad \forall i, \forall j > i, \quad (5.15)
\]

\[
\int v \left( \frac{d^2 \psi_i(v)}{dv^2} \right)^2 dv = 1, \quad \forall i. \quad (5.16)
\]

5.5 Implementation

The approach presented above is implemented in the MATLAB\textsuperscript{®} environment, along with two MATLAB\textsuperscript{®}-based graphical user interface (GUI) tools for enhanced simulation, testing, validation and visualisation.

The first tool implements functionality for deriving basis functions that are orthonormal with respect to a chosen surrogate objective function. The code uses symbolic mathematics for building the algebraic equations and supports the derivation of an arbitrary number of basis functions for spanning each direction of the parameterised patch space. A sample of a symbolically derived equations for the first 8 cross-stream basis functions is presented in Figure 5-2.

Figure 5-2. A sample snippet of the symbolically built first eight basis functions in cross-stream direction. (In the background: lines of MATLAB code, used for generating the equations).
Using symbolic implementation significantly increases the flexibility of the tool for testing different *surrogate* functions and facilitates further analysis, visualisation and validation by providing exact algebraic equations for the displaced surfaces and their first, second and third derivatives. The implemented code also supports derivation of basis functions that are *orthonormal* with respect to the change of the curvature of the patch in the stream-wise direction that is expected to lead to better convergence rates when optimising shapes for supersonic speeds (Sinnott & Osborne, 1961; Spreiter & Alksne, 1958). A screenshot of the main screen of the first GUI tool is presented in Figure 5-3.

![Figure 5-3. A screenshot of the first GUI tool, developed in MATLAB. Functionality for derivation of an arbitrary number of basis functions that are *orthonormal* with respect to a chosen *surrogate* objective function is implemented.](image)
Plots of the first 8 basis functions, calculated in the cross-stream and streamwise directions, are given in Figure 5-4 and Figure 5-5, respectively.

![Figure 5-4. Basis functions in cross-stream direction: a) even and b) odd functions.](image1)

As it can be seen from Figure 5-4, the first basis function (line 0, subplot (a)) can introduce a single “bump” in the patch rectangle that can be further controlled by adding more wavelets. For example, the next basis function (in terms of powers of $u$ (line 1 in Figure 5-4b)) has an S-shape and can be used to push the “bump” to one side, then functions with progressively more zeros can capture successively higher spatial frequencies in the solution. These higher frequencies will have relatively larger effects on the flow and so their amplitudes should remain smaller.

Similar observations can be made for the streamwise basis functions (Figure 5-5), but with the difference that they have smaller amplitudes, as their influence on the objective function is greater.

![Figure 5-5. Basis functions in streamwise direction: a) even and b) odd functions.](image2)
The second GUI tool developed for this research implements additional functionalities for calculation, exploration, visualisation, testing and validation of generated patch displacements. It allows calculation and visualisation of the resultant surfaces, as well as their first, second and third derivatives in the streamwise direction (Figure 5-6).

Figure 5-6. Basis functions exploration tool: a) generated displacement and b) streamwise curvature of the generated displacement.

The tool also supports manual exploration of the effects of varying a single or a set of design variables over the patch surface, as well as deriving some basic constraint conditions (e.g., for keeping the displacements in a given range or not allowing concavities). A screenshot of the main screen of the second GUI tool is presented in Figure 5-7.

Finally, although it is a job for the optimiser to find the optimal set of values for the design variables that minimise or maximise the objective CFD function, additional validation functionality is implemented for testing if the proposed displacement generation scheme can be applied for morphing a given stating surface, close enough to a desired test surface (e.g., to a one known to have better aerodynamic properties).
Figure 5-7. A screenshot of the second GUI tool, developed in MATLAB. A manual exploration of the effects of varying a single or a set of design variables over the patch surface is supported. Basic constraint calculation is implemented.

The latter question was of an initial concern, as the basis functions used in the approach are required to have both positive and negative curve regions (in order to comply with the constraints defined in their position, slope and curvature as outlined in Section 5.4.2), which was expected to result in a certain amount of oscillations (or “waviness”) in the designed shapes.

A sample morphing experiment is presented in Figure 5-8, where an initial wing surface (given in red), is perturbed to a desired test surface (with a major upward “bump”, given in green in Figure 5-8a). The result of the morphing (when using 8 basis functions in each direction) is presented in Figure 5-8b. It should be noted that although the resulting surface (received as a weighted sum of tensor products of orthonormal basis functions (given in green in Figure 5-8b)) might look uneven when zoomed, the oscillations are within 0.5 mm.
amplitude deviation of the desired surface (Figure 5-9). In the most cases, this deviation is insignificant, when compared to the distance between the nodes in the mesh grid, containing the investigated surfaces.

Figure 5-8. Wing surface perturbation: a) initial (red) and desired (green) shapes; b) fitted surface by 8 basis functions in stream wise and 8 basis functions in cross-stream directions.

Figure 5-9. Zoomed oscillations of the generated patch surface. Although the resulting surface (received as a weighted sum of tensor products of orthonormal basis functions might look uneven when zoomed, the oscillations are within 0.5 mm amplitude deviation of the desired surface. The units on the axis are meters.
5.6 Results

The applicability of the investigated approach is evaluated and validated by three main experiments, run under the same conditions, in terms of initial shape, CFD evaluation procedure, optimiser’s algorithm and settings, and computational resources.

The optimisation over the parameterised space is performed by employing a subplex method (a subspace-searching algorithm, based on Nelder-Mead simplex method (Rowan, 1990)) for minimising the total drag of an airplane geometry, where a rectangular wing area is selected for optimisation, as shown in Figure 5-1. On each iteration of the process, the value of the objective function is calculated by the commercial CFD solver and is then used by the optimiser to decide on what changes to introduce in the design variables (in order to reduce the overall drag of the current shape configuration). All the simulations are run on supercomputers (through an industrial partner) for 50 iterations and the values of the objective function are recorded. The experimental results are then evaluated and compared based on the achieved improvement over a given number of iterations.

The first experiment is for comparison purposes and investigates a case, where the discussed orthonormality exploitation approach is not applied. As it can be seen from the results presented in Figure 5-10, the optimiser hardly manages to introduce modifications that improve the objective function value. The optimisation starts from a total drag of the aircraft geometry of 219.02 drag counts (1 drag count equals to a coefficient of drag (CD) of 0.0001) and descends very slowly, achieving a minimum value of 218.95 drag units over 50 iterations.

In the second experiment, the proposed optimisation scheme is employed and the design space is spanned by 12 basis functions (3 in cross-stream and 4 in streamwise direction) that are computed to be orthonormal with respect to the streamwise curvature of the investigated region. The simulation results are
presented in Figure 5-11, showing much faster convergence than in the first case study. Starting from a total drag of 219.02 drag counts, the best configuration found over 50 runs has a value of 218.82 drag counts, which, in terms of improvement achieved over a number of iterations, shows about three times better performance. This is a strong indication that the theoretical concepts and assumptions followed in the investigated approach have practical grounds and can lead to much faster automated optimisation. It has to be noted that although a subplex optimisation algorithm is employed for this initial evaluation, conjugate gradient method is expected to lead to even better results (subject to further investigation).

Figure 5-10. CFD objective function convergence rate, when the investigated orthonormality exploitation approach is not followed.

Figure 5-11. CFD objective function convergence rate, when the investigated orthonormality exploitation approach is applied with basis functions, calculated to be orthonormal with respect to the streamwise curvature of the investigated region.

An interesting third experiment is also conducted for exploring the idea of how the optimisation process will be affected if, instead of spanning the space by basis functions orthonormal to the streamwise curvature, functions derived to be orthonormal with respect to the first derivative of the streamwise curvature
are used. This experiment tests to what extent the objective function is dependent on the change of the curvature in the streamwise direction. Although our expectations were confirmed for the investigated simulation conditions (specifically, optimising for subsonic speeds) and insignificant improvement in the convergence rate was achieved (Figure 5-12), it is included in this chapter as it is expected the situation to change, if the same heuristic is tested for supersonic speeds (as the objective function in this case is much likely to be dependent on the change of the streamwise curvature, as already discussed in Sections 5.2 and 5.3).

![Figure 5-12. CFD objective function convergence rate, when the investigated orthonormality exploitation approach is applied with basis functions calculated to be orthonormal with respect to the change of the streamwise curvature.](image)

5.7 Conclusion

An approach for automated optimisation of aerodynamic shapes, based on displacement patch generation using basis functions is investigated, implemented, tested and validated in this chapter. The method explores physics-based heuristics to derive sets of basis functions (for spanning the design space) that can significantly facilitate the work of the optimiser in finding better shape configurations. In particular, a surrogate objective function (related to the streamwise curvature of the perturbed area) is used for calculating the basis functions to be as close as possible to orthonormal with respect to a computationally very expensive CFD objective function.
Two MATLAB®-based GUI tools are implemented to support the calculation, exploration, testing and validation of the investigated approach. They are used to prepare and perform three major experiments for optimising real airplane geometry (being developed by an industrial partner) using a commercial CFD solver and run on transnationally located supercomputers.

Although additional simulations and tests are needed to better assess the capabilities of the investigated approach, the initial results are very promising in terms of improving the convergence rate of the computationally highly expensive CFD optimisation process and present opportunities for implementation of efficient automatic shape optimisation techniques.

The research presented in this chapter is conducted under a 6-month short-KTP (Knowledge Transfer Partnership) agreement between the University of Portsmouth and Airbus Operations Ltd. A summary of the investigated approach is presented as a short Case Study document in the Appendix section A4. In addition, a research paper ([1] in the List of Publications (Section 6.3)) is submitted to the Journal of Scientific Computing.
Chapter 6 Conclusion

6.1 Conclusions

Four large-scale real-world problems have been investigated and addressed in this research work. Several computational intelligence methods and approaches have been proposed, implemented, tested, validated and evaluated in the search of optimal and reliable solutions to several pattern recognition and process optimisation tasks. In their essence, the studied projects have been summarised below.

First, an intelligent visual inspection system for classification of textural images has been investigated. For each data sample, a set of descriptive features have been derived using spatial (grey-level co-occurrence matrices), filtering (Laws’ masks) and statistical (entropy) techniques. Approaches, incorporating supervised and unsupervised learning methods, have been considered and neural network based classifiers have been designed, trained and evaluated. The focus has been kept on the application of unsupervised non-linear dimensionality reduction techniques and unsupervised classification methods. An approach employing non-linear principal component analysis (NLPCA) in combination with self-organising neural networks has been proposed for solving the studied classification problem. A number of simulations have been designed, run and evaluated to test and validate the investigated approach and the received results have been critically analysed and compared.
Next, a problem for timely and reliable identification of radar signal emitters has been investigated. A number of experiments have been designed, implemented, simulated and evaluated for testing and validating the performance of the proposed intelligent systems for solving the studied classification task. Due to the large amount of missing values in the available dataset (i.e., values that could not had been fully intercepted or recognised), several techniques for dealing with the incomplete data have been explored, including listwise deletion and multiple imputation. A variety of data transformations techniques and neural network topologies have been investigated. Two broad case studies have been designed and executed for identifying two to eleven classes of radar signals. Very competitive performances of up to 87.3% classification accuracy rate have been achieved.

Third, an optimisation approach for improving the accuracy and efficiency of an industrial prediction model has been investigated. Its performance has been studied over a set of 16 specifically selected crude oil feeds. Global optimisation with genetic algorithms has been applied for optimising a number of model parameters and closing up the gaps between the predicted and measured data. Several MATLAB functions and a graphical user interface tool have been designed and implemented to assist the analysis, optimisation, testing and validation of the investigated model. Significant overall improvement in its performance has been achieved and the peak prediction discrepancies have been reduced to up to 20 ppm (parts per million) for the sulphur parameter and to up to 10 ppm for the nitrogen one.

Finally, a novel approach for optimisation of aerodynamic shapes based on displacement patch generation has been investigated, implemented, tested and validated. Physics-grounded heuristics have been used to derive sets of basis functions (for spanning the design space) that can significantly facilitate the optimisation process for finding better shape configurations. In particular, a surrogate objective function has been derived and used for calculating the basis functions, conditioned to be as close to orthonormal as possible with respect to
a computationally very expensive computational fluid dynamics (CFD) objective function. Two graphical user interface tools have been developed in MATLAB to support the calculation, exploration, testing and validation of the investigated model. Several experiments for optimising real aircraft geometry have been run on supercomputers through an industrial partner and opportunities for efficient automated shape optimisation have been highlighted.

### 6.2 Further Directions

Further investigation of the studied research topics can be made in the following directions:

- **Intelligent Pattern Recognition and Classification of Texture Images**: investigation of additional feature extraction (e.g., model-based) and image pre-processing techniques; enlarging the available textural dataset; further testing, validation and evaluation of the proposed approach; optimisation of the algorithm procedure and parameters; extending the work for 3D objects; examination of new application areas;

- **Intelligent Radar Signals Recognition and Classification**: investigation of additional statistical transformation techniques, such as principal component analysis (PCA), linear discriminant analysis (LDA), or non-linear principal component analysis (NLPCA); further research on techniques for handling missing data; varying some of the training parameters and exploring additional neural network (NN) topologies; extension of the study using unsupervised learning approaches (e.g., self-organising maps (SOM));

- **Intelligent Optimisation of Hydrotreating Process**: further optimisation of the model set-up parameters; additional testing, validation and
evaluation of the optimised model over a broader pilot- and commercial plant data; applying the same optimisation approach for improving the performance of other expert-based prediction models;

- **Intelligent Parametric Patch Perturbation for Aerodynamic Surface Optimisation**: additional research on the optimal number of basis functions to be used in each direction of the parametric patch; investigation of more candidates for basis functions; running evaluations over a variety of aerodynamic shapes; optimisation for supersonic speeds; extension of method for non-rectangular patches; introducing constraints during the optimisation (e.g., limiting the concave regions in the patch).

### 6.3 List of Publications

#### 6.3.1 Within the Scope of the Thesis

**BOOKS**


**JOURNALS**


CONFERENCE PROCEEDINGS


6.3.2 Outside the Scope of the Thesis

BOOKS

JOURNALS


CONFERENCE PROCEEDINGS

References


References


Appendices

A1. Intelligent Pattern Recognition and Classification for Texture Images

Figure A1-1. Samples of texture data from class “Beach”.

Figure A1-2. Samples of texture data from class “Corkstone”.

Figure A1-3. Samples of texture data from class “Desert”.

Figure A1-4. Samples of texture data from class “Lisbon”.
Figure A1-5. Samples of texture data from class “Pebble”.

Figure A1-6. Samples of texture data from class “Precision”.
Figure A1-7. Samples of texture data from class “Speckled”.

Figure A1-8. A screenshot of the developed and implemented GUI tool for the purposes of the first research topic: “Intelligent Pattern Recognition and Classification for Texture Images”. A number of functionalities for data pre-processing and unsupervised feature extraction, SOM classifiers training, results visualisation, comparison and analysis are implemented in the MATLAB environment.
Figure A1-9. Sample code snippets from the developed MATLAB GUI tool.
A2. Intelligent Radar Signal Recognition and Classification

Figure A2-1. Sample NN topology and training for Case Study 1. NN with 12 neurons in the input layer, 10 in the hidden and 1 neuron in the output layer is trained using Scaled Conjugate Gradient method. The training is stopped after six successive fails of the validation check.

Figure A2-2. NN learning sample distribution for Case Study 1.
Figure A2-3. Sample NN training performance for Case Study 1: training (blue), validation (green) and testing (red) errors.

Figure A2-4. Sample NN training error histogram for Case Study 1. The blue bars represent training data, the green bars – validation data, and the red bars – testing data. The errors have good Gaussian distribution.
Figure A2-5. Sample NN training state for Case Study 1. The training stopped after 88 epochs due to increase of the validation error in six successive checks.

Figure A2-6. Sample NN training Receiver Operating Characteristics (ROC) curves for Case Study 1. The curves (in blue) represent the true positive rate (sensitivity) versus the false positive rate (specificity). In the ideal case, the ROC plots would contain points in the upper-left corner, representing 100% sensitivity and 100% specificity.
Figure A2-7. Sample NN topology and training for Case Study 2. NN with 22 neurons in the input layer, 22 in the hidden and 11 neurons in the output layer is trained using Scaled Conjugate Gradient method. The training is stopped after six successive fails of the validation check.

Figure A2-8. NN learning sample distribution for Case Study 2.
Figure A2-9. Sample NN training performance for Case Study 2: training (blue), validation (green) and testing (red) errors.

Figure A2-10. Sample NN training error histogram for Case Study 2. The blue bars represent training data, the green bars – validation data, and the red bars – testing data.
Figure A2-11. Sample NN training state for Case Study 2. The training stopped after 93 epochs due to increase of the validation error in six successive checks.

Figure A2-12. Sample NN training Receiver Operating Characteristics (ROC) curves for Case Study 2. The curves represent the true positive rate (sensitivity) versus the false positive rate (specificity) for each of the classes. In the ideal case, the ROC plots would contain points in the upper-left corner, representing 100% sensitivity and 100% specificity.
A3. Intelligent Optimisation of a Hydrotreating Process

Figure A3-1. Excel interface to the hydprocessing prediction model. A number of crude oil and refinery reactor parameters are taken into account.

Figure A3-2. A screenshot of the developed and implemented GUI tool for the purposes of the third research topic: “Intelligent Optimisation of a Hydrotreating Process”. A number of functionalities for simulation, optimisation, visualisation, comparison and analysis are implemented in the MATLAB environment.
Figure A3-3. Sample code snippets from the developed MATLAB GUI tool.
Appendices

Figure 31: Reactor Inlet Temp. [°C] vs. Product Sulphur [ppm] for Nerefco LGO + 20%RHU

<table>
<thead>
<tr>
<th></th>
<th>53</th>
<th>54</th>
<th>55</th>
<th>Feed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measured Product Sulphur [ppm]</td>
<td>49.3</td>
<td>7.5</td>
<td>2</td>
<td>5313.94</td>
</tr>
<tr>
<td>Model Product Sulphur [ppm]</td>
<td>34.8</td>
<td>5.41</td>
<td>0.46</td>
<td>0</td>
</tr>
<tr>
<td>Optimized Model Product Sulphur [ppm]</td>
<td>41.53</td>
<td>5.89</td>
<td>0.54</td>
<td>0</td>
</tr>
<tr>
<td>Model Difference</td>
<td>15.5</td>
<td>2.09</td>
<td>1.54</td>
<td>0</td>
</tr>
<tr>
<td>Optimized Model Difference</td>
<td>7.97</td>
<td>1.61</td>
<td>1.46</td>
<td>0</td>
</tr>
<tr>
<td>Reactor Inlet T [°C]</td>
<td>335</td>
<td>355</td>
<td>375</td>
<td>0</td>
</tr>
<tr>
<td>Butyl Mercaptan [ppm]</td>
<td>1.03</td>
<td>0</td>
<td>0</td>
<td>217.59</td>
</tr>
<tr>
<td>Dibutyl Sulphide [ppm]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>Diisopropyl Sulphide [ppm]</td>
<td>0.23</td>
<td>0</td>
<td>0</td>
<td>602.7</td>
</tr>
<tr>
<td>Methyl Thiophene [ppm]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.9</td>
</tr>
<tr>
<td>Nonyl Thiophene [ppm]</td>
<td>0.03</td>
<td>0</td>
<td>0</td>
<td>237.72</td>
</tr>
<tr>
<td>Benzo thiophene [ppm]</td>
<td>0.03</td>
<td>0.01</td>
<td>0.01</td>
<td>4.37</td>
</tr>
<tr>
<td>Methyl Benzo thiophene [ppm]</td>
<td>0.01</td>
<td>0</td>
<td>0</td>
<td>388.58</td>
</tr>
<tr>
<td>Butyl Benzo thiophene [ppm]</td>
<td>0.02</td>
<td>0</td>
<td>0</td>
<td>519.11</td>
</tr>
<tr>
<td>Hexyl Benzo thiophene [ppm]</td>
<td>0.28</td>
<td>0.04</td>
<td>0.01</td>
<td>515.43</td>
</tr>
<tr>
<td>Octyl Benzo thiophene [ppm]</td>
<td>0.64</td>
<td>0.09</td>
<td>0.02</td>
<td>595.98</td>
</tr>
<tr>
<td>Undecyl Benzo thiophene [ppm]</td>
<td>0.26</td>
<td>0.03</td>
<td>0</td>
<td>307.6</td>
</tr>
<tr>
<td>Viglnyl Benzo thiophene [ppm]</td>
<td>0.2</td>
<td>0.02</td>
<td>0</td>
<td>309.53</td>
</tr>
<tr>
<td>DBT [ppm]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>44.23</td>
</tr>
<tr>
<td>Light Easy DBT [ppm]</td>
<td>29.56</td>
<td>4.73</td>
<td>0.45</td>
<td>609.32</td>
</tr>
<tr>
<td>Heavy Easy DBT [ppm]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>632.24</td>
</tr>
<tr>
<td>Light Hard DBT [ppm]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>68.21</td>
</tr>
<tr>
<td>Heavy Hard DBT [ppm]</td>
<td>7.08</td>
<td>0.73</td>
<td>0.03</td>
<td>159.92</td>
</tr>
<tr>
<td>Tetrahydro DBT [ppm]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Methyl Tetrahydro DBT [ppm]</td>
<td>1.72</td>
<td>0.19</td>
<td>0.01</td>
<td>0</td>
</tr>
<tr>
<td>Butyl Tetrahydro DBT [ppm]</td>
<td>0.15</td>
<td>0.01</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Pyridine [ppm]</td>
<td>0.04</td>
<td>0</td>
<td>0</td>
<td>51.26</td>
</tr>
<tr>
<td>Propyl Quinoline [ppm]</td>
<td>0.22</td>
<td>0.09</td>
<td>0.05</td>
<td>116.57</td>
</tr>
<tr>
<td>Pyrrole [ppm]</td>
<td>0.26</td>
<td>0.02</td>
<td>0</td>
<td>105.6</td>
</tr>
<tr>
<td>Ethyl Carbazole [ppm]</td>
<td>4.85</td>
<td>3.03</td>
<td>2.01</td>
<td>76.18</td>
</tr>
</tbody>
</table>

Figure A3-4. Sample screenshot of a report, generated automatically by the GUI tool for “Nerefco LGO + 20%RHU” crude oil feed. The optimised model MSE is reduced significantly.
Improving the process

The need

Hydrotreating processes are the most critical across all BP refineries for manufacturing clean fuel. In order to safely monitor and optimize the operation of its hydrotreating units, BP has developed a first-principle based kinetic simulation model.

However, as environmental regulation and market demand for clean fuel increases, BP faces new challenges. One issue is to accurately estimate the catalyst life length. Many short catalyst life cycles have been observed while producing ultra low sulphur diesel (ULSD). Analysis of these cases has shown that variations in the feed quality could have a significant impact on the catalyst life cycle (reducing it up to 30%), causing large financial losses.

The outcomes

As a result of this six month KTN Internship project, several major contributions have been made:

First, validation of a newly developed model was performed. It was compared to the existing model and tested to ensure there are no performance drawbacks.

Second, the existing hydropicrocessing model was linked to MATLAB®. This allows a large part of the analysis, optimization and reporting tasks to be semi or fully automated. In the future this code can be applied to different data sets and model versions.

Third, the existing model’s prediction capabilities were investigated and some model performance gaps were identified and analysed. A better insight into the model was gained as a result.

Fourth, model optimization was performed for some of the company’s most important prediction outputs. This led to a significant reduction in the gaps between the measured and predicted values for those parameters (Fig. 2).

Finally, a GUI including many of the functions developed during the course of the project was created. It will assist the company in further model analysis, optimization, testing and validation tasks on current and future BP models.

"The internship has produced new ideas about how to work and enhanced our understanding of the topic. The interaction between the project team and KTN has been particularly stimulating and we would welcome similar opportunities in the future."

Haitao Huang
BP Refining Technology, UK
Technical summary

The BP hydroprocessing model has the ability to predict the effects that different feeds have on apparent catalyst activity. It has played an essential role in the management of new hydroprocessing units across BP. However, with increasing standards and market needs for cleaner fuel, BP has experienced some problems with the capabilities of the model.

To investigate these problems, gap identification and analysis were run on the existing model.

To address these issues, model optimization techniques were investigated. Global optimization using genetic algorithms was performed for a set of model parameters and an improvement in the sulphur and nitrogen prediction capabilities was targeted. The code was implemented in a general framework so that the company could apply it to different models in the future—for optimization of predicted parameters and model improvement.

![Graphs showing model prediction capabilities.](image)

Fig. 1: Current model prediction capabilities: measured product vs. predicted product for sulphur (left) and nitrogen (right), for the Run68 dataset.

Figure 1 shows the prediction gaps identified for two of the parameters having greatest impact on the hydrotreaters’ operation - sulphur and nitrogen. Ideally, all the data points should lie on the diagonal line, which was not the case for the current model.

![Graphs showing optimized model prediction capabilities.](image)

Fig. 2: Optimized model prediction capabilities: measured product vs. predicted product for sulphur (left) and nitrogen (right) for the Run68 dataset.

As a result, BP’s hydroprocessing model prediction accuracy was improved and tools for further analysis and optimization were delivered to the company. A comparison between the available and optimized model capabilities is shown in Fig. 2.

“This internship has given me an invaluable insight of what the ‘atmosphere’ in a leading international company feels like. It has helped me to strengthen my mathematical, analytical and communication skills. I would recommend the KTN internship opportunity without hesitation.”

Nedyalko Petrov, University of Portsmouth

“Having some of our research investigations in the field of computational optimisation and related mathematical methods implemented for solving real-world problems from industry was really rewarding and stimulating.”

Ivan Jordanov
University of Portsmouth

This project was part of the programme of industrial mathematics internships managed by the Knowledge Transfer Network (KTN) for Industrial Mathematics. The KTN works to exploit mathematics as an engine for innovation. It is supported by the Technology Strategy Board, in its role as the UK’s national innovation agency, and the Engineering and Physical Sciences Research Council, in its role as the main UK government agency for funding research and training in engineering and the physical sciences.

---

Project Details

Partners
BP International Ltd
University of Portsmouth

Project investment
£21,000

Intern
Nedyalko Petrov

For details on the technology:
Haitao Huang
BP International Ltd
Haitao.Huang@uk.bp.com

For information on internships and other collaborations:
Lorcán Mac Manus
Industrial Mathematics KTN
lbmm@industrialmaths.net
+44 (0) 1483 565252

www.innovateuk.org/mathsktn

IP10-024 (January 2012)

Exploiting the power of mathematics
A4. Intelligent Parametric Patch Perturbation for Aerodynamic Surface Optimisation

Figure A4-1. Sample aerodynamic geometry. A rectangular patch is applied over a wing shape area, selected for optimisation. The arrows show the directions of the local surface normals.

Figure A4-2. A screenshot of the first developed and implemented GUI tool for the purposes of the fourth research topic: “Intelligent Parametric Patch Perturbation for Aerodynamic Surface Optimisation”. It supports functionality for deriving basis functions that are orthonormal with respect to a chosen surrogate objective function.
Appendices

Figure A4-3. A screenshot of the second developed and implemented GUI tool for the purposes of the fourth research topic: “Intelligent Parametric Patch Perturbation for Aerodynamic Surface Optimisation”. It supports manual exploration of the effects of varying a single or a set of design variables over the patch surface. Calculation and visualisation of the resultant surfaces, as well as their first, second and third derivatives in the streamwise direction is also supported.

Figure A4-4. Sample code snippet from the developed MATLAB GUI tools. In the foreground: symbolic equation derived for the generated displacement patch.
Figure A4-4. Sample code snippet from the developed MATLAB GUI tools. In the foreground: the symbolic equations derived for the phi and psi basis functions.
The need

Aerodynamic design engineers would like to have a tool that would help them to further fine-tune the aerodynamic parameters of a shape they have been designing.

Typical shapes for optimisation would include winglets, pylons, fuselages, etc., where the objective could be to maximise the configuration's lift force or to minimise the drag.

The designers would like to use it interactively with their manual design process, i.e. to receive results in a reasonable period of time.

The output shapes must be compatible with the design process (i.e. to comply with given constraints) and the results should be intuitive and easy to track and analyse.

The outcomes

As a result of this 6-month sKTP project, several major contributions have been made:

First, an approach for fine-tuning of aerodynamic shapes has been investigated, using basis functions selected to result in a very rapid convergence of the optimisation process.

Second, a software tool for calculating these basis functions has been implemented in MATLAB. It supports different kinds and number of basis functions and builds up the resulting surface equations and their first, second and third derivatives using symbolic representation (with no approximation). Additional testing and visualisation capabilities are also provided.

Third, an interactive GUI (Graphical User Interface) tool has been developed in MATLAB for exploration, visualisation and testing of a generated shape displacement field and its compliance with user-defined constraints.

Fourth, a Python implementation of the investigated approach is now available at the Company and initial tests have been started on supercomputers.

Finally, initial tests of the software within the optimisation process have shown the improvements in convergence that were hoped for from this approach. In a broader perspective, if this research can contribute towards new aircraft's drag counts reduction by at least 1 point, this can result in fuel savings of $1M/aircraft/year and will lead to proportional reduction in CO2 emissions.

The above mentioned contributions give good grounds for further exploration of the approach and present opportunities for a continued collaboration between the Partners.

"Nedyalko has done a fantastic piece of work allowing us, through this industrial partnership, to explore the generation of parametric shapes of specified surface patches on geometries that had previously been outside of scope. This work has been instrumental in determining methodologies to be employed in our overall process."

David Easterbrook, Airbus Operations Ltd

www.innovateuk.org/uk-skm
IP12-003 (December 2013) Exploiting the power of mathematics
Technical summary

When optimising part of an aerofoil, say the upper surface of a wing, the designer usually chooses a set of basis functions for small perturbations to the initial shape, and optimises over the coefficients of a perturbation that is a linear combination of those basis functions. However, the computational efficiency of the optimisation will be improved if the basis functions are approximately orthogonal, with respect to the objective function. In this project, the basis functions were chosen to be approximately orthogonal with respect to the aerodynamic considerations affecting lift and drag.

In this way, after using best practice to generate the initial design of the surface, and analysis to find the areas where it would be most beneficial to introduce changes, the design engineer can use the developed approach and software tools to optimise the design shape more rapidly.

The proposed approach has been investigated and several tools for exploration, constraints and orthonormality constants calculation, testing and perturbation visualisation have been built up in MATLAB. Finally, the algorithm has been implemented in Python and several tests have been started on supercomputers.

The initial observations, while running these simulations, show very promising results in facilitating the optimiser to move faster towards the desired objective function minimum (Fig. 3). For comparison purposes, results from a case where the orthonormality approach is not followed is given in Fig. 2.

Figure 1. Patch area selection for optimisation of a wing shape.

Figure 2. CFD Objective Function Minimisation Results when orthonormality approach is not followed (optimising for 15 variables).

Figure 3. CFD Objective Function Minimisation Results following the investigated orthonormality approach.

"This short Knowledge Transfer Partnership programme has given me invaluable experience in seeking solutions for large-scale real-world problems. It has helped me to strengthen my mathematical, analytical, programming and communication skills. I would recommend the IMSKTP opportunity to any PhD student without hesitation."

Nedyalko Petrov, University of Portsmouth

"This project has introduced us to a new model for optimization of aerodynamic shapes, based on displacement generation using basis functions, which was an important practical problem for AIRBUS. It was challenging project for Ned, but at the same time he has acquired insightful and beneficial experience in developing software tools for solving problems in a commercial environment."

Ivan Jordanov, University of Portsmouth

This project was part of the programme of industrial mathematics shorter KTPs managed by the Knowledge Transfer Network (KTN) for Industrial Mathematics. The KTN works to exploit mathematics as an engine for innovation. It is supported by the Technology Strategy Board, in its role as the UK’s national innovation agency, and the Engineering and Physical Sciences Research Council, in its role as the main UK government agency for funding research and training in engineering and the physical sciences.
A5. Word Cloud Thesis Summary

Figure A5-1. Summary of the thesis, presented as a word cloud.
A6. UPR16 Form

FORM UPR16
Research Ethics Review Checklist

Please complete and return the form to Research Section, Quality Management Division, Academic Registry, University House, with your thesis, prior to examination.

<table>
<thead>
<tr>
<th>Postgraduate Research Student (PGRS) Information</th>
<th>Student ID: 434051</th>
</tr>
</thead>
<tbody>
<tr>
<td>Candidate Name: Nedyalko Petrov</td>
<td></td>
</tr>
<tr>
<td>Department: School of Computing</td>
<td></td>
</tr>
<tr>
<td>First Supervisor: Dr. Ivan Jordanov</td>
<td></td>
</tr>
<tr>
<td>Start Date: February 2010</td>
<td></td>
</tr>
<tr>
<td>(or progression date for Prof Doc students)</td>
<td></td>
</tr>
<tr>
<td>Study Mode and Route:</td>
<td></td>
</tr>
<tr>
<td>Past-time</td>
<td></td>
</tr>
<tr>
<td>Full-time</td>
<td></td>
</tr>
<tr>
<td>MPhil</td>
<td></td>
</tr>
<tr>
<td>MD</td>
<td></td>
</tr>
<tr>
<td>PhD</td>
<td></td>
</tr>
<tr>
<td>Integrated Doctorate (New Route)</td>
<td></td>
</tr>
<tr>
<td>Prof Doc (PO)</td>
<td></td>
</tr>
<tr>
<td>Title of Thesis: Introducing Methods for Pattern Recognition and Optimisation</td>
<td></td>
</tr>
<tr>
<td>Thesis Word Count: 31852 (excluding ancillary data)</td>
<td></td>
</tr>
</tbody>
</table>

If you are unsure about any of the following, please contact the local representative on your Faculty Ethics Committee for advice. Please note that it is your responsibility to follow the University's Ethics Policy and any relevant University, academic or professional guidelines in the conduct of your study. Although the Ethics Committee may have given your study a favourable opinion, the final responsibility for the ethical conduct of this work lies with the researcher(s).

UKRIO Finished Research Checklist:
(If you would like to know more about the checklist, please see your Faculty or Departmental Ethics Committee rep or see the online version of the full checklist at: http://www.ukr.io/what-we-do/code-of-practice-for-research/)

a) Have all of your research and findings been reported accurately, honesty and within a reasonable time frame? YES

b) Have all contributions to knowledge been acknowledged? YES

c) Have you complied with all agreements relating to intellectual property, publication and authorship? YES

d) Has your research data been retained in a secure and accessible form and will it remain so for the required duration? YES

e) Does your research comply with all legal, ethical, and contractual requirements? YES

*Delete as appropriate
**Candidate Statement:**

I have considered the ethical dimensions of the above named research project, and have successfully obtained the necessary ethical approval(s)

<table>
<thead>
<tr>
<th>Ethical review number(s) from Faculty Ethics Committee (or from NRES/SCREC):</th>
<th>3CC6-10CE-0815-94E9-4F58-51B2-692C-A9C1</th>
</tr>
</thead>
</table>

Signed: (Student)  
Signed: (Student)  

Date: 04/12/2014  
Date:

If you have *not* submitted your work for ethical review, and/or you have answered ‘No’ to one or more of questions a) to e), please explain why this is so: