2012

An evolutionary optimisation study on offshore mooring system design

Zhuo Wang
University of Wollongong

Department of Engineering

An Evolutionary Optimisation Study on Offshore Mooring System Design

Zhuo Wang

"This thesis is presented as part of the requirements for the award of the Degree of Doctorate of Engineering of the University of Wollongong"

August 2012

ABSTRACT

This research was undertaken to provide tools to aid in the design and optimisation of offshore mooring systems. The tools present a set of feasible solutions with different performance characteristics. With an increased reliance on the exploitation of the reserves of hydrocarbon on marginal offshore areas, both in Australia and around the world, mooring mobile vessels becomes a more challenging task.

The latest evolutionary optimisation techniques offer new prospects for designing and analysing moorings, but to select an appropriate method of optimisation, both the classical deterministic search and evolutionary optimisation methods are reviewed. Genetic Algorithms (GAs) and Particle Swarm Optimisation (PSO) have been investigated in terms of their capability, strengths, and limitations. The GA and PSO have been benchmarked against each other, and PSO has been found to more efficient in the engineering applications considered.

Particle swarm optimisation was selected and implemented in the evolutionary mooring design computer software coded by the author. This software tool interfaces with the offshore industry commercial package OrcaFlex, which supplies results from model simulations. A case study of a mooring design is presented to demonstrate the ability of the optimisation tool, while the results provide potential improvements in mooring design solutions.

A performance based multi-objectives particle swarm optimisation (MOPSO) was developed where the multiple constraints can confidently be switched to problems with multiple objectives. Meanwhile, the results from the mooring optimisation obtained from MOPSO were compared with PSO, and showed that highly competing constraints leads the PSO search in an unhealthy manner. Alternatively, the performance based MOPSO design produces a set of solutions with a different performance which can be further processed and selected by offshore engineers. MOPSO offers greater flexibility and computational efficiency to optimise the final solutions.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank all the people who have helped, supported and cared about me while I completed this thesis and my degree.

First of all, I would like to thank my academic supervisor, Professor Tim McCarthy and industrial supervisor Dr Hayden Marcollo for their guidance, support and suggestions. Also, I would like to thank my sponsoring company AMOG Consulting Ltd., in Melbourne, Australia. They have provided both financial and invaluable technical consultation throughout the whole study. Sincere thanks to Australia-China natural gas technology partnership fund which provide me top-up scholarship. Without the help from them, I cannot finish this thesis.

Thanks to my adorable wife CeCe. She showed great patience and encouragement in helping me though difficulties. Last but not least, I have to thank my family who are thousands of miles away. Thank you all for your love, care, and support. I feel so proud.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY OF TERMS

AISC	American Institute of Steel Construction
ANN	Artificial Neural Network
API	American Petroleum Institute
BP	British Petroleum
CALM	Catenary Anchor Leg Mooring
DLL	Dynamic Link Library
DNV	Det Norske Veritas
FE	Finite Element
FPS	Floating Production System
FPSO	Floating Production, Storage and Offloading
GA	Genetic Algorithm
GUI	Graphical User Interface
IVA	Independent Verification Agency
LAT	Lowest Astronomical Tide level
LTP	Large Threshold Penalty
MCS	Multi-Constrained Search
MOPSO	Multi-Objective Particle Swarm Optimisation
MTP	Medium Threshold Penalty
NBF	Normalised Best Fitness in a swarm
NOF	Normalised global Objective Fitness values
NP	Normalised minimum Penalty
PSO	Particle Swarm Optimisation
QS	Quasi-Static design
RAO	Response Amplitude Operator
SA	Simulating Annealing
SALM	Single Anchor Leg Mooring
SCR	Steel Catenery Riser
SMS	Spread Mooring System
SPM	Single Point Mooring
STP	Small Threshold Penalty
TE	Tolerant Error

TLP	Tension Leg Platform
VIV	Vortex-Induced Vibration
WD	Water Depth

LIST OF SYMBOLS

A	unstressed cross-sectional area of the cable element
E	modulus of elasticity of cable material
H	the length of the horizontal projection of a cable element
k	the penalty threshold coefficient
L	the stressed length of a cable element
Lu	the unstressed length of a cable element
L_x	the horizontal scope of mooring line
L_y	the vertical scope of mooring line
P_1	the horizontal component of tension at node i
P_2	the vertical component of tension at node i
P_3	the horizontal component of tension at node j
P_4	the vertical component of tension at node j
P_H	the horizontal component of tension
P_V	the vertical component of tension
T	tension in the mooring line
t	time
V	the length of the vertical projection of a cable element
v	The velocity vector for PSO
w	the unit weight of a cable

1 INTRODUCTION

This thesis is concerned with the optimal design of mooring systems for floating offshore oil and gas production facilities. The optimisation scheme aims to reduce the amount of material used in the mooring system without compromising the levels of performance. An alternative way to look at this is to provide a range of solutions with different performance levels while choosing different sets of material design options. Either outcome can be achieved through a multi-objective evolutionary approach to design optimisation.

1.1 Background of the research

Offshore oil and gas operators are required to produce hydrocarbons in ever more inhospitable and deeper waters. The key to this development is the versatility of the modern range of floating offshore oil and gas production facilities. Floating production systems have advantages over fixed platforms. The latter become infeasible or uneconomical as the depth of water increases or the overall size of the hydrocarbon reserve becomes marginal. Offshore floating systems are less sensitive to the depth of the water and are reusable on multiple marginal oil or gas fields. One of the major challenges involved with the use of floating production facilities is to keep them on station, and to protect the valuable oil/gas risers from being over stressed by the vessels moving. If the mooring is too flexible then the risers will need to accommodate greater deformation, so an optimal mooring configuration will protect the vessel and the riser from extreme forces and displacements.

Current growth in oil and gas production relies on the exploitation of offshore reserves of hydrocarbon. This is particularly the case for Australia where large natural gas fields have been discovered and are currently being developed off the North Western shelf of Western Australia. Many of the newer oil and gas fields are smaller than previous developments, which mean that the production facilities must be designed with greater efficiency. As energy prices continue to increase, more of these marginal fields will become economically viable. This means that more floating production systems and more economical mooring systems will be needed, and with this increase in the use of marginal fields goes the need to re-use vessels and moorings. Since the vessels were originally selected for other locations, re-used

vessels may not be the best choice for the new location, so the challenge of mooring them in their new locations must be met.

1.1.1 Oil and gas energy in perspective

As a primary source of energy, the need for oil and gas is consistently increasing. It peaked in 2010, and the statistical figures for 2011 are still on their way. Figure 1.1 shows the world oil production over the last 45 years. This figure was released through the world economic outlook conducted by the International Monetary Fund (IMF) [1], and it is expected that world oil production will continue to increase.

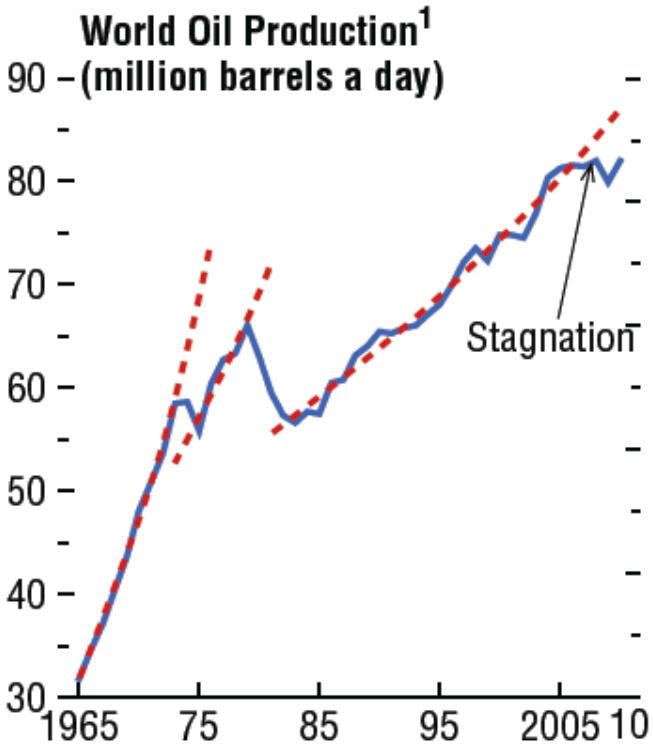

Figure 1.1 World Oil Production by IMF [1]

In terms of energy consumption, oil and gas play a vital role. Of the total amount of energy consumed in Australia at 2007-08, oil and gas accounted for 56 per cent [2]. BP's 'Statistical Review of World Energy'[3] reveals that consumption in 2010 overtook oil production by over 5 million bpd. BP's report highlighted the fact that the large difference between the production of oil and its consumption is due to alternative sources of fuel such as biofuels, oil from coal, and other non-conventional sourced that are not included in traditional oil production. Meanwhile, the increasing

demand for oil and gas is confirmed by statistics from the Central Intelligence Agency (CIA) USA [4]. In Figure 1.2, it can be seen that the increase in the rate of consumption exceeded the volume of oil produced, while BP also confirmed that the growth rate in the global consumption of fossil fuels was 3.1% in 2010 [5].

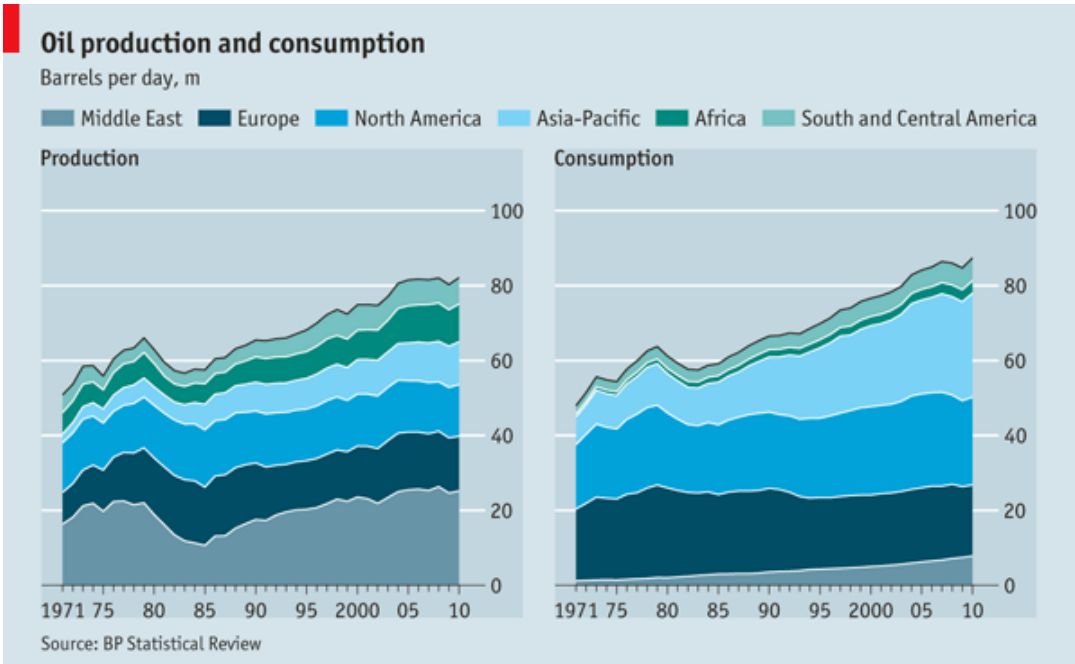

Figure 1.2 Comparison of oil production and consumption [3]

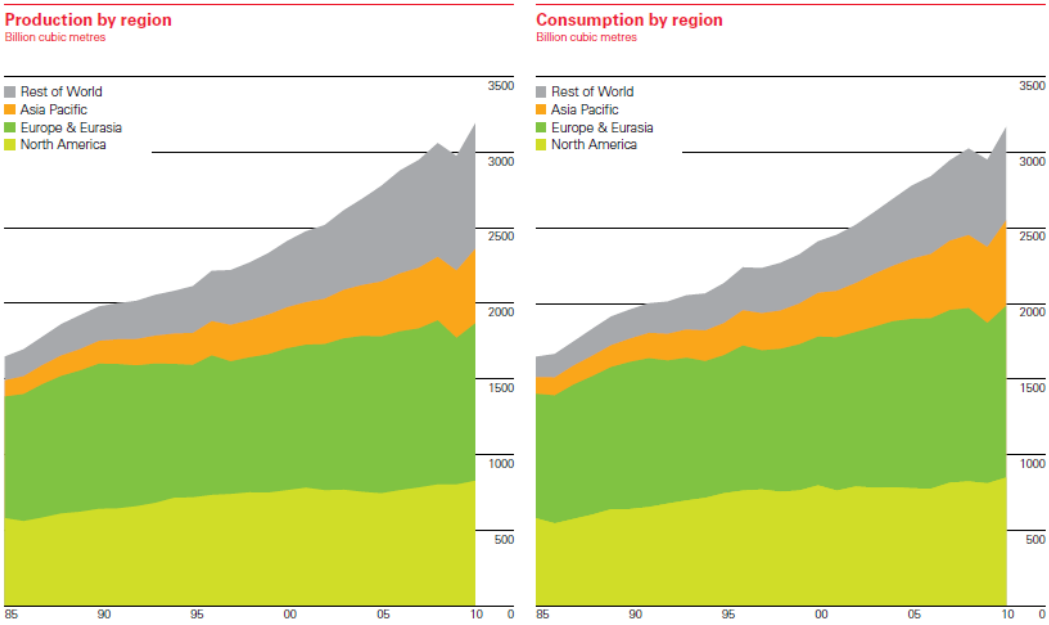

Figure 1.3 Comparison of gas production and consumption [6]

Figure 1.3 shows the production and consumption of gas from 1986 to 2010. Over the last 25 years the production and consumption of gas shows a generally steady pattern of growth, and this steady trend can be back dated to more than 50 years ago. Therefore, before any greater substitute source of energy other than oil is found, oil and gas will continue to play an increasingly important role in both economics and in our lives.

1.1.2 Importance of the offshore sector in the production oil and gas

The increasing use of hydrocarbon products has stepped up the demands for greater outputs of petroleum and extensive exploration. In 1947, when the first offshore steel structure was erected in the Gulf of Mexico [7], the world entered a new era of offshore oil exploration, and this offshore exploration for hydrocarbons has been expanding into ever deep water since then.

In Figure 1.2, total oil production in 1989 was about 63 million bpd. Of this, about 15 million bpd were produced offshore, which covered 24% of this total production. By 1997, the production of offshore oil increased to 21 million bpd out of a total of 75 million bpd, and the percentage stepped up to approximately 28%. In the same year, the gas produced from offshore fields was about 20% of total gas production. As a matter of fact, over the previous two decades, offshore production is the main contributor to the growth of oil production, since onshore production has plateaued.

The offshore sector plays an even more important role in Australia's petroleum industry. According to the Department of Resources, Energy and Tourism (RET) Australia [2], about 95% of Australia's petroleum production is from the vicinity of offshore sedimentary basins. Especially in offshore Liquefied Natural Gas (LNG) production, ranked as the world's 18^{th} largest natural gas producer, Australia had exported about 18 million tonnes of LNG in the year of 2009, which made it the world's 4^{th} largest LNG exporter. Of all the figures listed above, new facilities such as floating LNG (FLNG) and new improved drilling equipment and techniques, keep Australia as 'state-of-art' in the offshore industry.

1.1.3 Challenges for offshore moorings and engineering

There are two major components in a floating structure offshore oil exploration and production system, as shown in Figure 1.4. One is the topside or superstructure such as fixed or mobile platforms, which mainly operation above sea level, and the other is offshore pipeline structures such as moorings and risers that are below sea level. These are discussed in detail in Chapter 2.

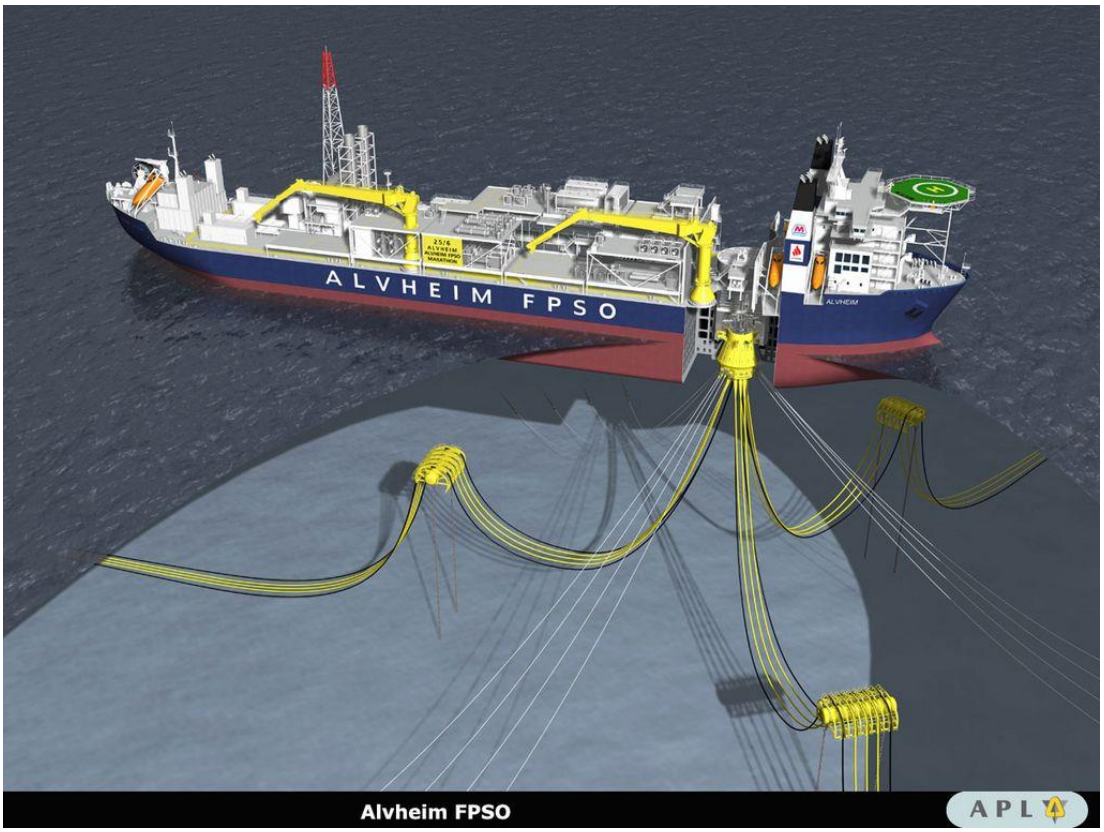

Figure 1.4 Offshore FPSO oil exploration and production system [8]

A mooring system is used to keep a floating vessel or structure in position through fixed objects such as piers, a quay, or the seabed. A riser is essentially a conductor pipe which connects vessels, platforms, and/or floaters on the surface to wellheads at the seabed to convey fluids such as hydrocarbons, gas, mud, water, and sometimes mechanical equipment. In order to handle the harsh sea environment, the mooring system must be developed for keep the floating vessel on station so that risers can be protected from excessive forces and allow it to function properly.

As the main part in stationkeeping of offshore vessels, the mooring systems must be designed strong enough so that it has sufficient stiffness to stop the vessel drifting an excessive distance away from the station. Meanwhile, it also must be designed to be flexible enough to allow sufficient compliance to avoid overload in the legs or anchors. Along with these design requirements, a large number of design variables and constraints such as mooring patterns, material selection, mooring leg lengths, mooring sizes and pre-tension levels increase the complexity of mooring design.

Offshore oil exploration and production has more challenges than onshore, and they stem mainly from two sources. The first is the loads applied to offshore structures, and the second is constructing complex industrial facilities in remote and hostile environments.

Apart from normal loads such as wind, earthquake, and service loads, offshore structures are exposed to wave conditions that involve substantial risks. Around the world, some of the most extreme environmental loads on structures result from wind storms and associated waves, while the occasional passage of extreme hurricanes in the Gulf of Mexico can also be dangerous for offshore platforms. Extreme storm conditions in the North Sea can be expected at a frequent rate, and similarly, parts of Australia's offshore waters are subject to some of the most violent sea conditions known. In all, offshore structures need to be designed to withstand all of these harsh situations.

The difficulties begin with installing the facilities, followed by the practical limitations of lines clashing, and material fatigue, and etc. Since floating production facilities are much more dynamic, all the components need greater attention in their design compared to onshore.

Although the offshore oil and gas industry relies on cutting edge technology, the complexity and large dimensions of the system makes full scale experiments inconvenient to carry out. The current design methodology for selecting mooring and riser configurations for floating oil and gas production facilities relies heavily on the expertise of the engineers and a trial and error approach [9]. Sometimes, even with

experienced engineers, it is difficult to achieve workable or efficient designs in the given timeframe, and even when workable designs are found, there is often little time available to improve or optimise these solutions.

1.1.4 Evolutionary computation in offshore optimisation

Inspired by natural evolution and adaption, evolutionary computation uses the ideas from biological or social behaviour in the study of computational systems [10]. Compared to traditional computational systems that are good at accurate and exact computation, evolutionary computation can handle situations where inaccurate, noisy and complex information are included.

Some problems are difficult to solve because the designer is uncertain where to begin the solution search [11]. Such a situation is common in optimisation of offshore engineering projects. The complexity, explored in section 1.1.3, optimisation in offshore field of research is one of the challenging tasks that cannot be easily tackled by traditional optimisation techniques. In evolutionary optimisation, the search process begins with randomly generated candidates which minimise the efforts of acquisition of pre-knowledge. Then they narrow down the trajectory towards to near global optima. Therefore, the application of evolutionary optimisation algorithms to offshore engineering benefits the circumstance where solutions range is normally unknown beforehand. The computational cost of simulation is also considered expensive. The evolutionary optimisation algorithms require minimal simulations and meanwhile maximise the performance.

1.2 Research question

The background of this research gave general information about offshore oil and gas engineering. It revealed the challenges and necessity to optimise mooring systems in view of the increasing demand for FPSOs around the world and in Australia. It also provided the motivation to conduct this research. The sentence that describes the research in question form is,

How to apply evolutionary optimisation techniques to offshore mooring design in a rational manner?

To solve a doctoral level research question, it was suggested that the general question be sub-divided into smaller questions [12]. Thus, the main research question becomes:

- What are the driving parameters in the design of an offshore mooring system?
- How are these parameters interrelated?
- What is the most suitable evolutionary optimisation method for offshore mooring design?
- Is it feasible to optimise complicated problems in the mooring design?
- What is the limit to the complexity that can be optimised?

1.3 Aims and objectives

The ultimate goal of this research is to develop methodologies and tools for the design and optimisation of offshore mooring cable structure systems. These improvements include a tool to allow engineers to better assess and manage performance in offshore engineering design. The proposed system also promotes a more efficient use of structural materials, so the aim here is to integrate existing software tools that are used to design marine moorings within an evolutionary optimisation algorithm.

The specific objectives of this work are to:

- Become familiar with offshore terminology, design concepts, and design practices such as the corresponding design codes of API [13] and DNV [14];
- Understand the knowledge related to design methodologies and analysis for mooring systems;
- Critically analyse the behaviour of offshore mooring systems to identify the chief design parameters and constraints that govern their design;
- Obtain a suitable case study of mooring design;
- Examine and benchmark the technologies of optimisation technologies and locate the appropriate application for this optimisation;
- Create new software tools and methodologies for the design of offshore mooring structure systems;

- Demonstrate the capabilities and potential improvement the new tools and methodologies might deliver for the multi-criteria optimisation of moorings.

The main focus of this research is the design of marine mooring structures. This study is concerned with capturing the design method and optimising the flexible solutions. The proposed new method is able to accommodate a range of design alternatives, allowing engineers to choose between competing solutions. For example, a mooring can be optimised to reduce top tension whilst maintaining station keeping parameters. Alternatively, the amount of material can be optimised while keeping the top tension within a desired range. By providing a tool that can rapidly produce a range of design options, the engineer can better control the overall process.

1.4 Arrangement of the thesis

As a reading guideline, the remaining chapters of this thesis are structured as follows:

- Chapter 2 reviews the two group of optimisation techniques implemented in engineering. Since the traditional search and artificial intelligence search approaches are gaining current popularity, they are included to give a better understanding of how they can be used to optimise offshore structural designs.
- Chapter 3 introduces essential components of offshore floating platforms, moorings, and material as well as examples of real world offshore applications. An overview of the offshore commercial design package OrcaFlex is also given. Meanwhile, the terminology used in the offshore discipline has also been introduced throughout this chapter. It provides a structural analysis and design methodology for offshore cable structures. This chapter includes reviews of offshore mooring design requirements, and the standard design processes. It also shows a practical industrial design flowchart of a mooring design. Apart from all these above, an improved approach to iterative flexibility has been developed and documented.
- Chapter 4 describes the close investigation of components, operations, and techniques of genetic algorithm (GA) and particle swarm optimisation (PSO). Mathematical characteristics with regards to its stability and convergence

have been studied and PSO related work in engineering applications have also been overviewed. The chapter finishes with multi-objective particle swarm optimisation (MOPSO), including a detailed technology for searching for the global optimum and handling constraint issues.

- Chapter 5 compares two of the artificial intelligence optimisation approaches, PSO and GA, in a benchmark manner. This chapter presents the benchmark in purely mathematical functions and an engineering application. Various components of these two approaches have not been looked into, but the improved findings are drawn to the following chapters in the optimisation of offshore cable structures.
- Chapter 6 describes the application of PSO and MOPSO and to optimise marine mooring structures. It gives an overview of the optimisation software that incorporates Matlab and OrcaFlex and introduces a new method in offshore design, performance based design method.
- Chapter 7 includes prints the solutions and results from performance based design. This method shifts constraints to several design targets. Meanwhile, all relevant information of interest with regards the project can be listed simultaneously. The new design method gives offshore engineers better solutions with more flexible options. A case study based on the optimisation of a mooring is presented in this chapter and the result provided by the PSO and MOPSO with the new design method is compared. The final results reveal the benefits of using the new method in the area of offshore design and optimisation.
- Chapter 8 concludes and summarises the whole thesis. It provides a thorough summary of the whole projected carried out during this research. Meanwhile, original contributions made by the author are illustrated. It finishes the research project by making recommendations for future research in this area.

1.5 Project context statement

This research project is a joint collaboration between AMOG Consulting Pty Ltd., Melbourne and the University of Wollongong in Australia. AMOG is a global provider of solutions to the energy, resources, defence, rail, and maritime construction industries. It specialises in a few engineering fields, including:

engineering design & analysis, risk assessment & safety engineering, infrastructure support & upgrade, installation engineering & support and legal & expert witness.

The research project was also sponsored by the Australia-China natural gas technology partnership fund through a top-up scholarship. Apart from the desire to develop a long term mutually beneficial partnership in the energy sector between Australia and China, the fund also has an interest in increasing knowledge of offshore engineering and technology, and applying it the field of natural gas.

1.6 Research methodology

The design of mooring systems requires a fundamental understanding of the technology associated with, and the design methods used in the offshore engineering industry. The optimisation of such designs necessitates a fundamental understanding of both classical and non-derivative optimisation techniques. The main themes of research in this current work can be depicted in the mind map shown in Figure 1.5.

The familiarisation of the terminology used in offshore mooring engineering, and the mathematical prerequisites in related areas of this research, served as footsteps towards further study of mooring analysis and design. Meanwhile, it is necessary to create new software tools to facilitate the design of moorings and cultivate the capability of coding and programming of any software package used in this research because optimising moorings demands real world experience of mooring from the offshore oil and gas industry.

All the activities illustrated in Figure 1.5 can be categorised into two groups in terms of knowledge, explicit knowledge and tacit knowledge, respectively. Explicit knowledge can be captured through literature reviews such as review articles, advanced level books on related topics, published journals, proceedings of conferences, and sometimes workshops, etc,. [15]. However, tacit knowledge such as experience is contextual and released spontaneously, it cannot be fulfilled through classical learning methods such as reading textbooks or journal papers, and reviewing books [16].

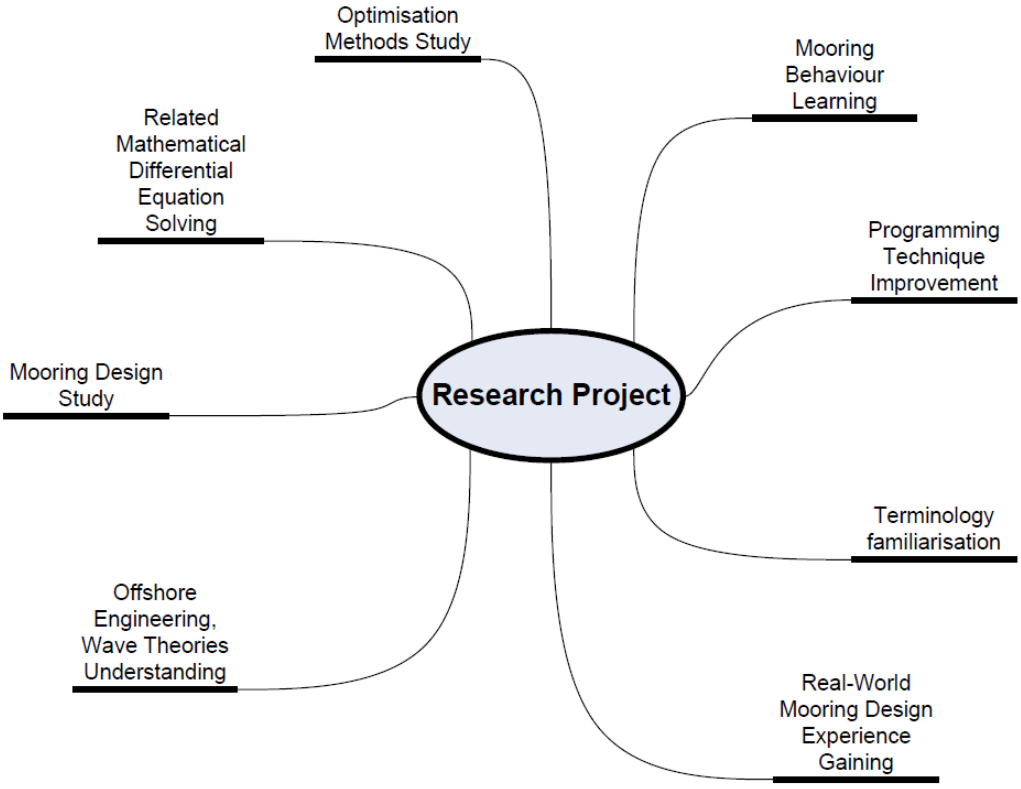

Figure 1.5 Research project activities

A comprehensive review of both classical and evolutionary optimisation techniques has been conducted to search for the most suitable algorithm that can be used to optimise moorings. However, reviewing optimisation methods alone is not enough to find the one method suitable for mooring design. A comparison between potential optimisation methods is deemed to be a good method for searching for the final result. Hence, the procedure for selecting the most suitable method is based on the benchmark of optimising potential candidates from the review. A benchmark of the optimisation algorithms should be taken within the mathematical functions and engineering applications. The most suitable algorithm should perform well when searching for near global optimum and superior computational efficiency in terms of finding the solutions. Furthermore, it should also be easy to manipulate and simple enough to integrate into mooring designs. In a tangible manner, the selected optimisation method requires a relatively small number of coefficients.

The commercial offshore software package OrcaFlex was used in this research project because the offshore industry believes it is the leading software package available on the offshore dynamics market, and its simulation accuracy has been proved by many offshore projects. Similar packages such as Flexcom, AQWA, ABAQUS, ANSYS and ARIANE have their particular strengths in finite element simulation, but OrcaFlex performs well, and it is versatile and efficient in offshore marine systems. In order to manipulate this software proficiently, training is vital, and although self training with the aid of a software manual is a traditional method, it is not good enough. Professional training in the use of this advanced software should be given by qualified parties. A week long OrcaFlex (in Oct 2010 at Perth, Australia) training course was taken along with an OrcaFlex user group meeting.

Computer programming was identified at the end of the first internship (end of 2008) as an essential prerequisite. The frequent use of Matlab in the industry from observation and internship made it the programming tool for this study. Matlab is taken to be the coding language due to its popularity and ease of communicating with third party software in engineering applications. Due to the popularity of Matlab, many commercial software packages have developed interfaces for further communication. Meanwhile, a large number of optimisation toolboxes embedded in Matlab is another important reason for incorporating it in this research. The optimisation toolbox minimises efforts from the user side and the programming skills required to design an optimisation tool for a mooring system are gained through training and other activities. The other extra benefits of incorporating Matlab into this research are the level of communication developed between OrcaFlex and Matlab through the dynamic link library. Details and an example of applying this dynamic link library can be found in Appendix A. Courses on the use of Matlab were delivered by MathWorks (end of 2009), and the University in related topics such as advanced programming skills and optimisation, etc., were taken during the periods of this research.

Experience is a type of knowledge, like tacit knowledge, that involves learning and skills that cannot be written down or recorded, and cannot easily or effectively be gained from explicit methods of acquiring knowledge. Gaining experience requires

practice, discussion, and working with experienced offshore engineers in real cases and environments. To fit into this type of learning environment, an internship in related areas like offshore mooring and riser design and analysis is beneficial. To obtain both explicit and tacit knowledge, as suggested by Wasonga [16], an internship in the offshore industry is a way to build outcomes. Therefore, an internship in the offshore industry is deemed to be an integrated way to obtain the tacit knowledge type of experience [17].

In addition to the normal academic research work undertaken at UOW, six months were spent working (in 2 3-month internship) in offshore engineering at the head office of AMOG Melbourne. Sponsorship by AMOG has many benefits; first, AMOG is a consulting company that has a number of worldwide offshore experts who are able to deal with difficulties of offshore designs, and second, it has access to real design data based on 20 years of experience in the industry, so many of the practical limitations and difficulties in design methods are well understood. The focus of this research is therefore to provide and improve flexible solution options that can easily be monitored. Third, the case study data provided by AMOG are on the basis of actual projects, even though some details are omitted due to confidentiality. However, the concept and principle of optimisation on offshore structures are well demonstrated. A brief description of the principal industrial work undertaken for the sponsoring company is listed in Appendix B.

This internship in the offshore engineering industry was conducted in two stages; the first stage commenced at the end of 2008, and the second stage commenced at the end of 2009 and was completed by the beginning of 2010. This arrangement was based on an understanding of the cognitive domain of knowledge. As shown in Figure 1.6, understanding begins at the lowest level and proceeds through increasingly more complex and abstract levels, with memory of the knowledge gained, forming the base [18]. An understanding of knowledge is built on the foundation of remembering. A comprehensive review of offshore structures and technology was considered to be prerequisites before undertaking the first stage of internship, hence it was not carried out at the start of the research before a general literature review of offshore structures was taken.

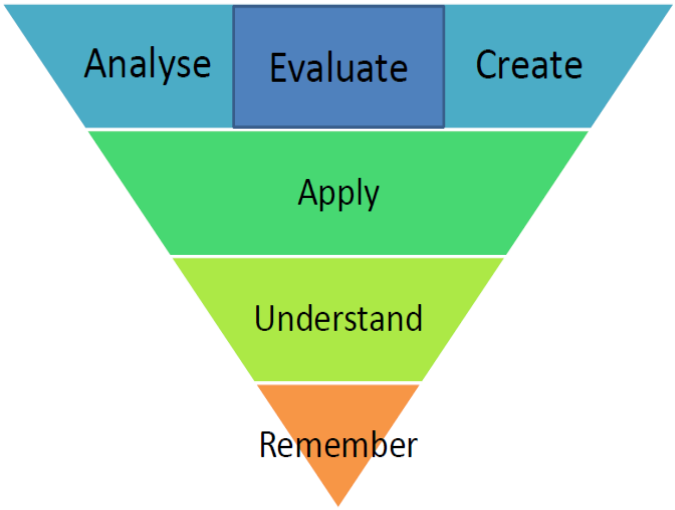

Figure 1.6 Categories in the cognitive domain

The purpose of the first internship (three months) was to become familiar with the terminology and monitor general offshore design activities, which resembles the two bottom levels of the reverse pyramid. A gap between the first and the second internship consolidated the knowledge and further research after exploring the real world of offshore oil and gas. The second internship focused more on offshore designs.

The second three month internship focused on more advanced levels of the cognitive domain, as shown in Figure 1.6. This involved gaining an insight into mooring designs, access to real design data, the opportunity to apply simulation software to real projects, and opportunities to talk to offshore engineers about developing new design methods, etc. A full list of all the methods mentioned in the research methodology have been summarised in Table 1.1, and correspond with the requirements of knowledge and achievements.

Table 1.1 Knowledge requirements and achieving methods table

Require knowledge and skills	**Obtaining methods and achievements**
Knowledge of offshore system and design standards	Literature reviewed has been accomplished with related to offshore structures. Work experience with AMOG Consulting Pty Ltd., Attendance of related seminars and conferences
Offshore oil and gas terminology and design codes, and familiarisation practices	General exploration and reading of literatures, design practices and codes. Working experience. Discussion with academic and industrial supervisors and peers during internship.
Offshore theories and structural theories	Previous learning outcomes by author. Literature reviewed with regards to the offshore theories and structures. Consultation with supervisors and industrial supervisors.
Analysis of offshore mooring structure systems	Literature reviews. Publications by author. Discussion and consultation with academic supervisors.
Selection of suitable optimisation algorithms	Literature review of optimisation methods (including traditional and evolutionary category). Benchmark studies of potential optimisation candidates from reviews.
Offshore mooring cable structure optimisation	Literature review. Benchmarking and comparison of optimisation method. Consultation with academic supervisors.
Software skills	Self-training according to user manuals. Special training courses taken by author. Working experience through placements. Discussion with peers and professionals during internship.
Validation of optimisation results of case study	Discussion with academic and industrial supervisors and sponsoring company. Literature review. All sorts of methods listed above.

1.7 Review of methodology

The research methodology has categorised this research project as applied research rather than basic research. According to Rajasekar [15], applied research has an outcome which has immediate application to related fields, whereas this project was undertaken in a close link with an offshore industry company to obtain a tacit knowledge of mooring design. To ensure this research project has a certain depth of optimisation of mooring systems, the internship at AMOG for this applied research was helpful because this link with the offshore oil and gas industry was of practical use in these research activities.

Every component of the research topic has been examined in terms of feasibility and workability. The methods required to obtain the required information and knowledge have been discussed. The methodology illustrated a systematic way of support to solve the problem logically. One of the advantages of dividing the final aim into components and objectives is to provide targets in the progress and tangible evaluations so that achievements can be monitored step by step [12]. The strength of the research methodology assures that the research project will be completed, while the progress made during the research can easily be checked.

The method of literature review and other traditional learning methods from explicit knowledge were considered insufficient in terms of having experience with real world mooring designs. This is due to the essence of this type of knowledge because first, a literature review of experiential knowledge can be limited in its sources since experience is based more or less on a summary of the length of time spent working in a specific area, not a type of knowledge can be clearly reasoned and documented. Second, due to commercial confidentiality issues in offshore engineering at its market, the skills and experience gained have been treated as valuable information that is not accessible to the public. The confidentialities and patent agreements have kept certain highly competitive and exclusive entrepreneurs in their services and products. The best way of gaining experience is in the environment, working, discussing, and practising. Therefore, gaining a working experience of this type of valuable information is much better. The methodology part reviewed the difference between explicit and tacit knowledge so that different methods have been designed to

gain specific knowledge. Meanwhile, understanding the cognitive domain made the internship more valuable when it was supported by learning the theory aspects of this industry.

From a methodology point of view, AMOG is only one offshore consulting company, so their experience of offshore may be biased to some extent. Admittedly, AMOG has gained its offshore experience after more than twenty years, and a border contact of the offshore industry would give more generous ideas and experience in mooring design. However, considering that all the engineers at AMOG are from around the world, their advice and opinions are typical of offshore areas. Meanwhile, the scope of the research project on the offshore industry can be extended since there is no doubt that a generous contact with the offshore industry would have fulfilled this research project much better. Moreover, given the timeframe required to finish this PhD research, only a six month placement at AMOG was undertaken by the author. In all, the methodology of this research has arranged a systematic way to arrive at a solution for this research question, that the logical is reasonable, and methods delivered are feasible.

2 OPTIMISATION TECHNIQUES

2.1 Introduction

This chapter reviewed several of optimisation techniques in two major categories. The first group of optimisation is classified as a traditional search, and the other category is defined as an artificial intelligence search. This categorisation of optimisation techniques is distinguished by the way they search for better positions. The classical methods are based on a calculation of the gradient to find the optimum solution, whereas the artificial intelligence methods are stochastic in nature, with an extended suitability.

2.2 Traditional search

Hill climbing, gradient methods, mathematical optimisation, and some other local methods are categorised as traditional approaches, but in terms of optimisation, a common feature of all these methods is the ability to find a better answer under the circumstance of being given an initial solution. However, the better answer they provide is heavily reliant on the quality of the initial solution. In situations where the survey of objective functions has been inadequate, or there is insufficient information in the response spectrum, the problem can hardly be optimised without some assistance from other strategies.

Within a limited range of the given starting position, a traditional search can locate local optima in a smooth and continuously functioning landscape, but different traditional search methods have different characteristics. From a purely mathematical model solving an engineering application, every search method can be applied in according to the applicable situations.

2.2.1 Hill climbing

Hill climbing is one of the famous optimisation techniques in the family of tradition searches. It can be mathematically described as: minimising a function $f(x)$, where x is a vector. Adjusting a single element in the vector of x and re-assessing the function $f(x)$ at each iteration, such that any changes that improve $f(x)$ are accepted as the new solution.

Hill climbing is an iterative optimisation technique where the initial position is first given, and then it is repeated for the vicinities until a better position has been found. This algorithm stops when it can find no better solution. Therefore, the chance of finding a real global optimum is highly dependent on the quality of the initial position. Fenwick [19] claimed that this can be overcome by repeating the solution process from different initial starting positions. The random restart hill climbing method mentioned here is known as Shotgun hill climbing.

Since the vector of *x* in hill climbing optimisation is adjusted on the basis of a single element, evaluating the objective functions can be extremely arduous for problems with plenty of variables. Nevertheless, if a near optimum position can be initially provided, the hill climbing techniques can be applied for the post fine tuning process. Fenwick [19], Michalewicz [20] and Beighlter, et al [21] have demonstrated the application of hill climbing in the tuning process associated with other optimisation techniques.

2.2.2 Gradient methods

Of all the gradient methods, for example, the Euler method and gradient descent method, etc., Newton's method is a typical and the most widely used example [22]. One starts with a randomly selected initial guess with a reasonable acknowledgement of a function, and where the approximation of this function is represented by a gradient, and then the interception of this tangent line to the *x* axes is calculated. The interception point serves as the root approximation of this function, where the process can be iterated for convergence.

Mathematically, Newton's method can be expressed as a continuity and differentiable function *f(x)*, where the simple algebra of the solution in one variable is

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{2.1}$$

where $f'(x_n)$ is the first order differentiation $f(x)$ at the point of x_n.

The first order derivative information is used to determine the direction of the search, although if available, the second order derivative can be used to speed up the rate of convergence [23]. Meanwhile, Newton's method can also be applied to approximated numerical methods, such as the example in section 3.6. With an increasing complexity in dimensionality and discontinuity, etc., the difficulty with Newton's method increases because his method may converge on an inaccurate solution, or even diverge, if the objective functions are discrete. One of the limitations according to Onwubiko [24] is that the method tends to locate local optima in cases with great complexity.

There are a few practical considerations while using Newton's method, which include:

- How difficult it is to calculate the derivative
- The speed of the function convergence
- What if poor initial estimates are made
- What if the method fails to converge
- Any further techniques needed for over shoot mitigation

2.2.3 Brute force search

A brute force search, also called an enumerative or exhaustive search, is a trivial but general problem solving technique that enumerates each and every solution in the search space until the best answer is found. The reason for the brute force search gaining in popularity and generality is its simplicity [25]. Only three procedures need be implemented; the generating solution candidate, the evaluate solution candidate, and the valid solution candidate. If it exists then finding the optimum solution to a problem by a brute force search can always be guaranteed since it exhaustively searches every possible candidate.

However the limitations of the brute force search is obvious, it may require centuries of computational time to enumerate and validate all the possible solutions because the size of a potential solution for a real world problem is enormous. For example, to solve the classic problem of the 50-city travelling salesman in an optimisation field, the brute force search needs to go through 3×10^{64} potentially different routes before

concluding on the final solution. Even equipped with the most advanced computer, the computational time for such a huge search cannot be estimated.

2.2.4 Greedy algorithms

The concept of greedy algorithms assume that the acumination of the local optimum at each stage leads to the global optimal. This is also why it is called the "greedy" algorithm. Greedy algorithms complete problems in a series of steps where one best piece at a time makes the best available decision. However, it is clear that the optimum decision at each separate step does not necessarily return the optimum global solution. As a result, Michalewice [25] summarised that greedy algorithms normally pay for their simplicity by failing to provide good solutions, especially in complex problems with multiple parameters.

2.3 Artificial intelligence search

2.3.1 Tabu search

In 1989 Glover [26, 27] introduced a mathematical optimisation method based on trajectory techniques. This search algorithm is an iterative search that utilises flexible memory structures to store visited solutions in order to prevent any revisiting. Essentially, a tabu search has been modified from the hill climbing method described in section 2.2.1, with the ability to escape from local optima.

The fulfilment of escaping from local optima is through the acceptance of a neighbouring solution, whether or not it is better than the current one. In this method the acceptance criteria is called the 'tabu list'. Tabu lists contain a number of strategies such as forbidding and freeing, as well as intermediate and long term strategies, to determine which solution should or should not be stored [28]. With the aid of the tabu lists, this method is guided in the direction desired. Even though a tabu search has algorithms that can avoid local optima, the capacity to escape them is claimed to be limited.

2.3.2 Artificial neural networks (ANNs)

Inspired by biological neural networks, an artificial neural network (ANN), or numeral network (NN), is a numerical mathematical model that processes

information through interconnecting nodes (called neurons). Composed of layers and neurons, from inputs to outputs, information is processed through the network architecture with great adaptation and learning [29]. A typical ANN system has a layer of input neurons, followed by one or more hidden layers, and then a third layer of output neurons, as shown in Figure 2.1.

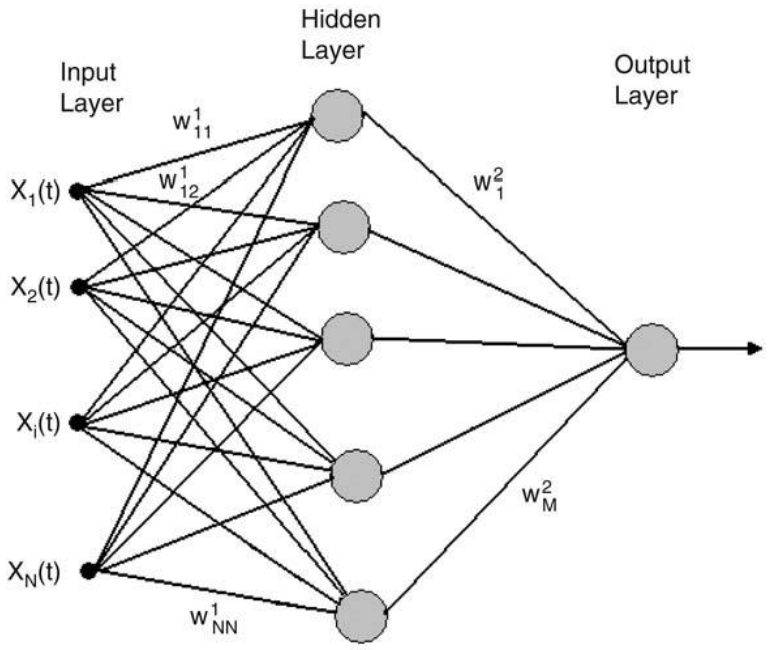

Figure 2.1 Schematic view of neural network architecture [30]

The outstanding advantage of ANN is the link, or ‘mapping’ between the input and output response, that can be built without any understanding of a problem. This technique requires little knowledge of the problem itself because the intermediate layers can be fully tuned by comparing the inputs and outputs. The tuning procedure here is called ‘learning’. The purpose of learning is to set the appropriate weights of the entire network in order to minimise errors between the outputs and actual target results. A more detailed overview of the engineering design in ANN can be found in Rafiq’s [31] work. Furthermore, another prominent advantage of ANN application is that less computational effort is required when the ‘training’ has been accomplished. Apart from all those features, Pham [32] summarised the fact that ANNs have other major features such as tolerance to noise and damage to components.

Due to differences in the variety of problems, an ANN approach has to repeat the learning process case by case, i.e., individual training is required for each situation in a problem to give accurate results. An ANN application to offshore cable structures may be built and trained for each line and for each load. According to Guarize [30] et al, building algorithms that automatically optimise ANN architecture instead of having to manually train in offshore applications by trial and error would be very handy.

2.3.3 Simulated annealing (SA)

Inspired by annealing in metallurgy, simulated annealing is a technique based on the characteristic of atoms to tend towards finding lower internal energy when they become unstuck from their initial position after being heated. This method was first introduced by Metropolis et al [33] when modelling the equilibrium state of collected atoms in 1953. However, the first use of simulated annealing in engineering optimisation was back 1983, by Kirkpatrick [34].

SA is an iteration based algorithm where a small displacement is given to an atom at each iteration, and the change in energy in the system is then calculated. If the change in energy in the new position is negative, the new position is accepted, otherwise acceptance depends on the probability in terms of the temperature and other combined constants. The pseudo-code of SA can be presented as follows:

WHILE the temperature is low enough (stopping criteria reached)
    FOR each atom, generate randomly chosen neighbours for a given state
        FIND the smallest energy in the neighbourhood
        IF the energy less than the current energy
            SET the atom to the lowest energy state
        ELSE
            Give probability on the basis of the current temperature
        END
        Change the temperature according to the cooling schedule
    END
END

At the starting stage the SA tends to accept a worse move because the temperature is high and the ability to escape local optimum is high. With the cooling schedule, the algorithm is focused more on the exploration of local minima [35].

2.3.4 Hybrid algorithms

The hybridisation of different optimisation algorithms can overcome some limitations due to their nature, while their strengths can also be capitalised. According to Preux [36], the hybridisation of optimisation can be categorised into sequential models and parallel models respectively.

Sequential hybrid methods always use a couple of optimisation techniques. One intelligent method searches for a near optimal solution region while the other local search method can be used to fine tune the true optimal solution. Burke and Smith [37], and Cunliffe [38] both incorporated a local search into a genetic algorithm, whereas Galinier and Hao [39] presented an evolutionary algorithms framework with a tabu search.

Parallel hybrid methods have two algorithms that work either simultaneously or interactively. Premalatha and Natarajan [40] combined the standard particle swarm optimisation (introduced in Chapter 4) with the simulated annealing (SA). They demonstrated with benchmarking functions that with the SA to diversify the positions of the particles, a combined optimisation showed a greater rate of convergence and a better quality solution. Schmidt and Thierauf [41] incorporated a genetic algorithm (introduced in Chapter 4) based optimisation, called differential evolution, with a threshold accepting algorithm which resembles simulated annealing (SA), to form a combined heuristic optimisation technique. They proved the efficiency of the new combined technique with some simple engineering applications. Unlike the single optimisation approach, the parallel hybrid approach often makes use of the strength of one algorithm to make up for the limitations of the other. This means that an improved algorithm can be formalised with a combination in parallel with two or more methods. Sometimes, however, this combination

requires a complex coding process to integrate different techniques into a single algorithm

2.4 Chapter summary

This chapter looked closely into the different search and optimisation techniques in terms of two categories, while also presenting the strengths and limitations of those techniques. The traditional search techniques can locate local optima in a smooth and continuously functioning landscape, while the artificial intelligence group can learn and be trained from the experiences of previous candidates using pre-set rules that tend to provide better solutions than local optima.

Traditional search techniques attempt to deduce a better position for the solution in analytical way by using a function derivation. Those gradient search methods can be useful and efficient in smooth and continuous functions, but when 'spikes', noises, break downs, or discontinuities occur, this type of search method has its limitations.

On the other hand, the artificial intelligence optimisation methods are deterministic and stochastic in the way they search for optimisations. Here each individual is evaluated in its original function, which has avoided the way of calculating derivations. There are a few ways in which stochastic searches advance classical searches; superior information from previous candidates can be inherited to offspring by several pre-set rules, and the searching history and knowledge can also be shared among individuals for the purpose of optimisation. The artificial intelligence methods are thus considered to be robust when handling unfavourable response surfaces, and they can also find global optima in harsher search situations.

3 OFFSHORE MOORING DESIGN & ANALYSIS

3.1 Introduction

This chapter presents an overview of floating platforms and a structured review of offshore mooring systems. The mechanics and analysis of mooring line for design and simulation are illustrated in details. An improved method of flexibility iteration in analysing mooring lines is introduced with a verification example. This method has proved to be efficient at calculating catenary mooring cables.

In addition to the structural mechanics, this chapter also reviews design standards, codes of practise, and the processes involved in designing moorings. A mooring design flowchart of the offshore industry based on real world engineering practice has been concisely summarised. The characteristics of mooring design have been examined for the purpose of further optimisation. This chapter finishes with an example of a mooring design. This example explains the complexity and characteristic problems inherent in mooring designs, and reveals the common design variables in this type of problem. Meanwhile, the potential and necessity of optimising moorings has been demonstrated with an illustration of the example.

3.2 Floating and tethered platforms

Floating platforms can be generally grouped into ‘Permanent’ and ‘Mobile’ facilities [42], which are relative concepts with regards to design life. The ‘permanent’ facilities are designed to be moored in place for 20 to 30 years. Therefore, this group of platforms have to withstand extreme environments such as 100 year weather conditions. ‘Mobile’ facilities are used for drilling, construction, and installation. They both have great applications in a range of water depths. In the following, different types of floating and tethered platforms are discussed.

3.2.1 Tension leg platforms (TLPs)

A tension leg platform is a vertical moored floating offshore structure that normally consists of hulls and topsides. The three main configured components are the pontoons, stability columns, and the deck. The entire hull is anchored to the sea

floor by a series of vertical tendons. Due to its high axial stiffness, the vertical, pitch, and roll motion of platforms are negligible. The first TLP was actually built in relatively shallow water in the mid-1980s. Since then, the use of this type of platform has extended to moderate and deep water. Vertical tendons, as the most critical part of the whole structure, play a significant role in keeping the platform safe and stable. Most commonly, tendons consist of high grade steel tubes with the wall thickness increasing with the depth of water. All the vertical tendons are connected to the foundation templates which in turn, are fixed by driven piles. Meanwhile, all the vertical tendons are always in tension as the pontoons generate an excess air buoyancy force. Figure 3.1 is a schematic diagram of a TLP. TLP technology was not serving Australia at the time the author finished this thesis, but it is expected that TLPs will be delivered to browse for LNG development in Oslo and Perth, Australia.

Shell's Auger TLP is a typical example; the foundation templates are 872m the below the sea level, and the hull is composed of four 22.6m diameter circular cylindrical columns welded to four 8.5×10.7m box pontoons. It is installed in the Garden Banks area 214 miles SW of New Orleans. The platform consists of a 91m^2 deck which supports drilling rig production facilities, and accommodation modules, etc. There are three vertical tendons connected between each cylindrical column and foundation template, and a total of 12 tendons 847m long that hold the platform in place. The tendons are connected to the platform by mechanical connectors which are specially designed to minimise fatigue in the welds. The tendons were manufactured from X60 grade steel with a 33mm wall thickness. The total project was finished at a cost of US$1.2 BIL [43].

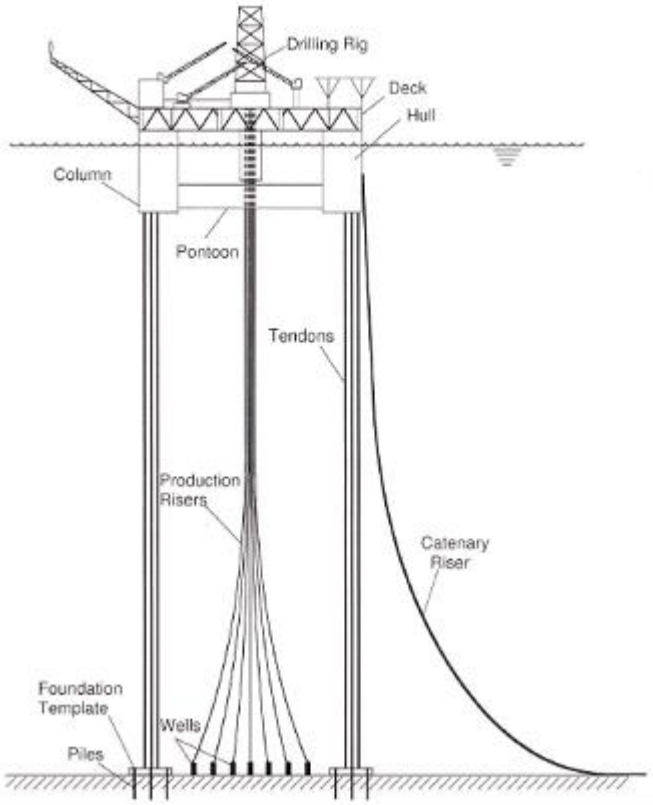

Figure 3.1 Schematic diagram of a tension leg platform [44]

3.2.2 Spar platforms

A spar is a deep draft floating cylindrical caisson, as shown in Figure 3.2, which relies on conventional moorings to maintain its position. This means that vertical tendons are not necessary. Compared with TLPs, spar platforms can be described as a TLP with just one large diameter cylindrical column serving a dry deck on the top. The cylindrical column has a huge diameter, and when a floating structure has such a huge diameter, it is very hard to keep it stable in the water, even though the deep draft spar produces very favourable motion characteristics. Engineers have worked out that putting ballast in the spar stabilises the structure in a still environment. Most floating structures are designed to correspond with the wave loading. In some cases, transportation and installation loading controls the design of the spar. Spars are built and transported horizontally, and then upended onsite in water. The wave loads during transportation and buoyancy loads during upending can be a critical factor in their service life.

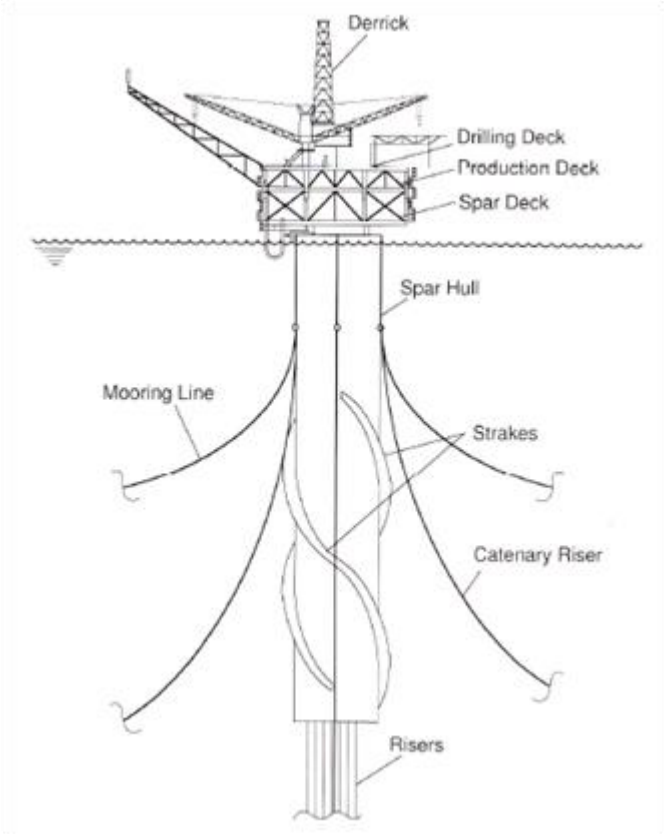

Figure 3.2 Schematic diagram of the Hoover-Diana spar platform [44]

One of the famous and largest spar platforms is the Hoover-Diana project located approximately 160 miles south of Galveston, Texas [45]. The Hoover-Diana platform has a 215m high hull, a diameter of 36.6m, and weighs 35000t. The total cost of the project was about US$1.6BIL. The platform is held in place by 12 mooring lines with a 2011m total length of wire rope and 300m of chain anchored to the sea floor.

One of the most common problems seen on a spar platform is the Vortex-Induced Vibrations (VIVs) caused by passing currents. This phenomenon resembles water running through piers under a bridge deck. A vortex can easily be found in the downstream direction of piers. Unlike bridge piers, spar platforms are not fixed to the sea floor and so vibrations are always inevitable, and although the VIV resulting from currents can be reduced by adding extra strakes (also called spoilers) to the spar they also have a negative effect by increasing the drag force on the spar.

Up to 2003, there were 14 spars in service or under construction around the world [42], but since the Spar and TLP have favourable heave and pitch motions, they are the only floating platforms that have been used for dry trees.

3.2.3 Semi-submersible floating production systems (FPSs)

At first glance, the schematic view of the FPS in Figure 3.3 suggest that it looks similar to a TLP, in that it also consists of hulls made of buoyant columns and pontoons, and has a deck placed on top of the structure. But a closer examination reveals that instead of using vertical tendons to hold the structure in place, it is actually held in position by the mooring system.

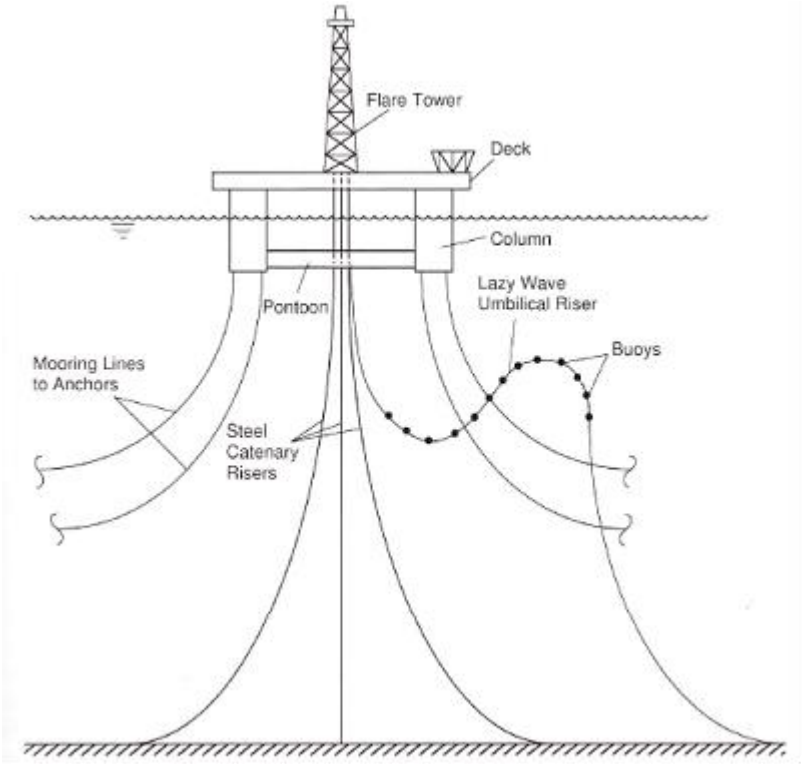

Figure 3.3 Schematic diagram of a FPS [44]

The Na Kika semi-submersible hull, located in the Mississippi Canyon (140 miles SE of New Orleans) in the Gulf of Mexico, is a typical FPS consisting of four buoyant columns with pontoons moored in 1936 metres of water supporting four topsides modules [46]. The columns are 17.2m square and are fixed by 16 semi-taut catenary mooring lines. The pontoons have a cross section of 12.4×10.5m. All of these give the steel hull a combined weight of 20,000t. It is very convenient method of

converting from deep water drilling vessels, and since FPS's are more flexible than TLP's, their loading and storage ability are normally restricted by a certain scale.

3.2.4 Floating production, storage and offloading facilities (FPSOs)

Floating production, storage and offloading facilities receive and store crude oil in tanks inside their hulls until it can be offloaded to shuttle tankers or transported to shore. FPSOs are normally shaped like ship, whereas other semi-submersible type hulls with storage, or cylindrical hulls such as the Sevan Voyageur, can also be seen but are very rare. A schematic of an FPSO system is shown in Figure 3.4. They do not usually house drilling rigs and other related facilities, and it is the only type of rig that does not process gas. Gas is either re-injected into the reservoir or sent to other gas storage facilities.

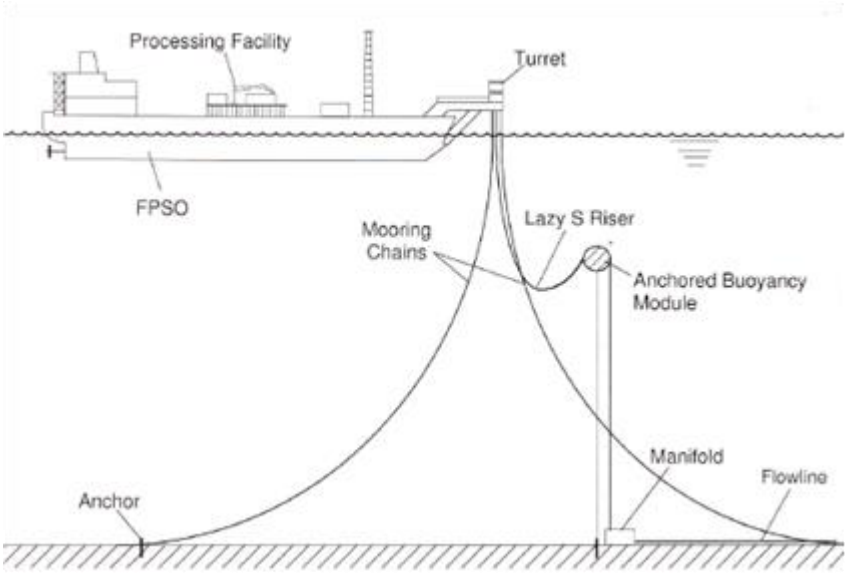

Figure 3.4 Schematic diagram of a typical FPSO system [44]

FPSOs are very versatile and as such are used in remote areas where no pipeline infrastructure exists, or in areas where local conditions make connection to a coastal facility risky. When extreme load cases occur, the ship can even drop off all the risers and auxiliaries to escape the storm. To drop all the attached facilities, a special turret has been designed and is usually assembled at the bow of the FPSO's. Two major problems are solved by the special revolving device; the first is the problem of

the risers twisting as the ship rotates, and the second is that the lower part can be abandoned when extreme weather conditions are forecast.

The weather vane is also a very helpful piece of apparatus in the extreme weather conditions that frequently occur in regions such as the North Sea. A semi-weather vane mooring, which is a spread mooring with slack moorings aft, has been used for the Brazil rig to give the vessel an option of limited weather vaning [47]. In benign weather areas such as West Africa and South East Asia, the FPSO can be in spread in a moored pattern.

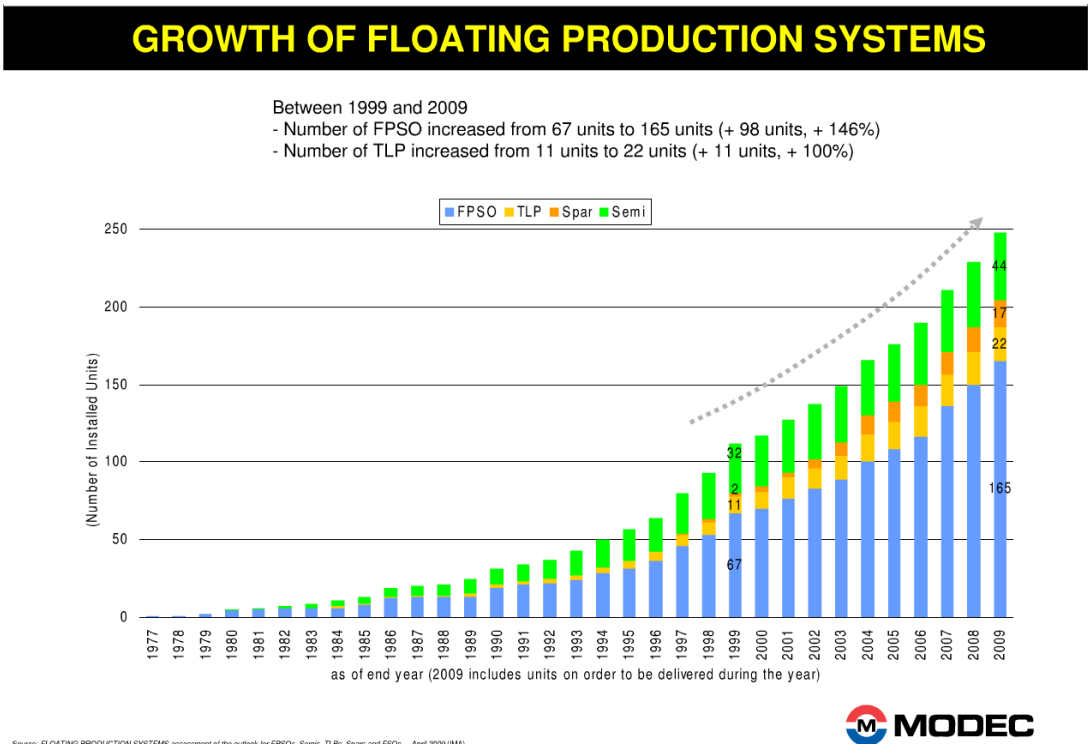

Figure 3.5 Assessment of the outlook for FPSOs, Semis, TLPs, Spars and FSOs [48]

There has been a wide use of FPSOs around the world. According to Shimamura [48], and as shown in Figure 3.5, this increase in FPSOs reached 146% from 1999 to 2009. In the FPSO markets, the North Sea and Brazil are the primary users [42], although Asia and Australia are also a very important market for FPSOs. As of July 2009, there were 15 FPSO projects operating in Australia, with more than half of them operating in water 100 to 300m deep [48]. The main reason for the boom in FPSO's in West Africa, Brazil, South East Asia, and Australia is that they are best

fitted for marginal oil fields, which is why this type of floating platform is of primary interest to the Australia offshore industry.

Kyriakides and Corona [44] described a typical example of offshore FPSO in Angola. The Girassol offshore system consists of a permanently moored FPSO for the production of oil from 23 wells covering an area over 10 × 14 km, and it was installed 210 km NW of the capital, Luanda. This FPSO is 300m long, 60m wide, has a displacement of 396,000t, and can store up to 2MMbbl of oil. To retrieve the oil, gas is re-injected into the reservoir and the oil is transferred to a transport tanker from an offloading buoy located 2km from the FPSO. The total investment for the development by TotalFinaElf was $ 2.8 billion.

3.3 Mooring system components

To have all these floating platforms fit for purpose, mooring systems must be designed, and as an important part of the offshore infrastructure, these designs are essentially a trade-off between a system compliant enough to avoid excessive external forces and a system stable enough to avoid damage to the risers from intolerable offsets.

A vessel is defined as being moored if it is fastened to a fixed object such as a quay, a pier, or the seabed, and sometimes to a floating object such as an anchor buoy. This activity is always with the assistance of ropes, chains, or hawsers which are called mooring lines. Mooring lines associated with a floating platform form a fundamental system whose primary purpose is to keep the floating structure in position, within limits. DNV-OS-E301[14] is the comprehensive code covering the technical provisions, certification, and classification of mooring systems. Other standards detailing station keeping systems are API-RP-2SK [49] and ISO 19901-7 [50]. A brief review of mooring systems, including the mooring material, anchors, and types of mooring systems are covered in the following sections.

3.3.1 Mooring material

Steel chains, wires, and synthetic fibre are the most common materials used for mooring lines. The selection of one type of material over the other two, or a

combination of materials, depends on type of structure to be moored and the relative depth of water. Details can be referred to in section 3.9.

3.3.1.1 Chain links

In chain mooring fabrication, two types of chain rings are available: one is chain link without a stud (studless), and the other is chain link with a stud. Chain links with a stud and without a stud can be seen in Figure 3.6. The stud link chain has proven to be strong and relatively easy to handle when used for mooring MODUs and FPSOs in shallow water. The studs also provide extra stability to the link, both in servicing and handling. The stiffness of the chain links is closely associated with the nominal dimension of diameter. More details, including other accessories such as Kenter shackles, pear links, and C-links can be found in DNV-OS-E301 [14].

3.3.1.2 Wire ropes

Steel wire rope sections can be of various types of construction, as shown in Figure 3.7. According to DNV-OS-E301 [14], six strand wire ropes are the commonly seen ropes for mobile offshore units, while spiral strand wire rope performs better in fatigue and corrosion, and is therefore the rope most commonly used for floating production units designed to stay in position for a long time.

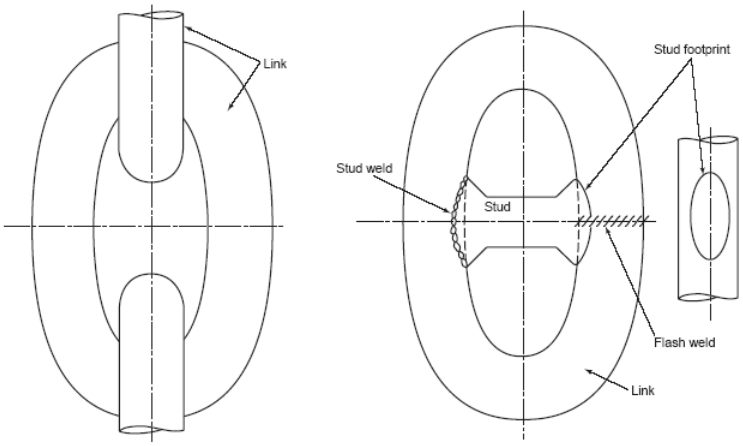

Figure 3.6 Studless Chain link & Stud Chain link [51]

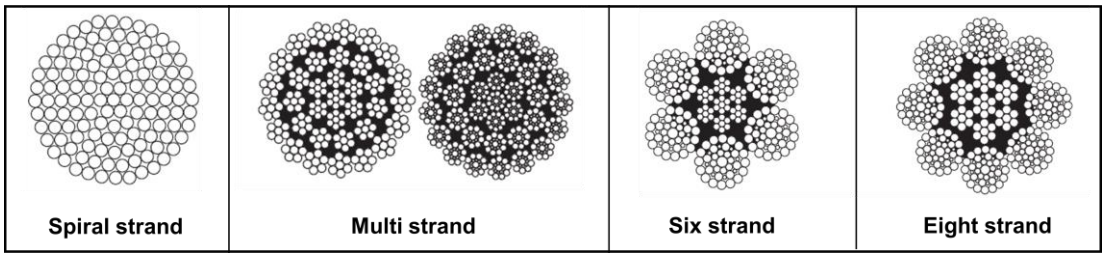

Figure 3.7 Steel wire rope construction types [14]

3.3.1.3 Synthetic fibre ropes

Yarns supplied from fibre makers are the materials used to make synthetic fibre ropes. The core rope contains lubrication, fillers, or other material. Plastic or rubber sheet form the outer protective jacket of the synthetic fibre rope, as can be seen in Figure 3.8.

Figure 3.8 A picture of synthetic fibre rope [51]

Synthetic fibres currently being considered for use in mooring system include [14]:

- Polyester (polyethylene terephthalate)
- Aramid (aromatic polyamide)
- HMPE (high modulus polyethylene)
- Nylon (polyamide).

There are a number of advantages to using synthetic fibre rope. It can be well adapted to deep water applications since its mass is much lighter than chain links. It also has lower stiffness, but its visco-elastic behaviour could bring some challenges when simulating the mooring system in computer package. However, durability is

one of its main weaknesses compared to steel chains, and the tension cycling of this material is not as good as other types [51].

3.3.2 Mooring anchors

In this section the equipment used for mooring is discussed. Anchors with flukes, plates, suction pile, and gravity types, are the most frequently seen equipment used in permanent moorings. Anchors of different types are discussed in the following paragraphs.

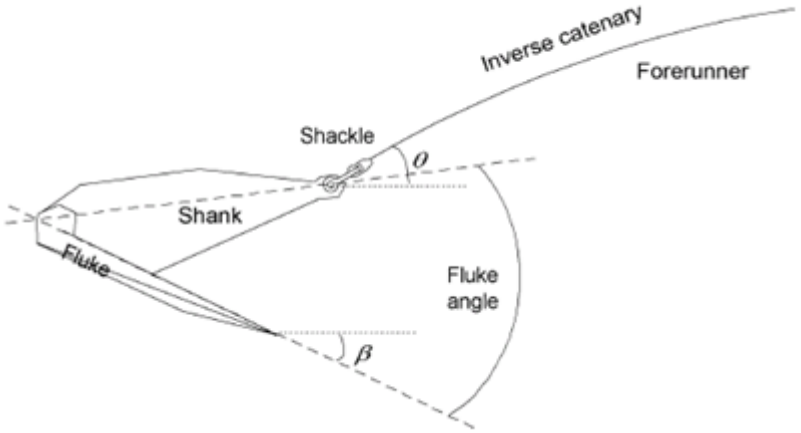

Figure 3.9 A fluke anchor [14]

A fluke anchor, also called a drag embedment anchor, normally has a fixed angle ($\theta+\beta$ seen in Figure 3.9) ranging from 30 to 50 degrees. The holding force of the fluke creates tension in the mooring position system. According to DNV-OS-E301 [14], lower angles are normally used for sand and/or stiff clay, and larger angles are used for soft or consolidated clay because the flukes penetrate the sea bed deeper and provide better resistance. As the anchor is dragged along the sea bed, it digs in below the bed.

Gravity anchors are designed to use their self weights against uplift as well as displacement violation. DNV-OS-E301 [14] clearly states that the capacity of gravity anchors should not be higher than the submerged weight of the structure. Another type of anchor commonly seen is the suction anchor. Relevant design requirements for these anchors can be referred to in DNV-OS-C101 [52].

3.3.3 Types of mooring systems

Generally speaking, the station keeping system for a floating structure has two major categories: the first is a single point mooring (SPM), and the second is spread mooring (SMS). As a complementary technology of stationkeeping, the dynamic positioning (DP) stationkeeping can be used solely or to assist a catenary mooring.

3.3.3.1 Spread mooring

Spread moorings are mostly used for semi-submersibles and spars. It consists of a group of mooring lines that terminate at the corners of a vessel to hold the vessel's heading. Figure 3.10 is a schematic view of a spread mooring system.

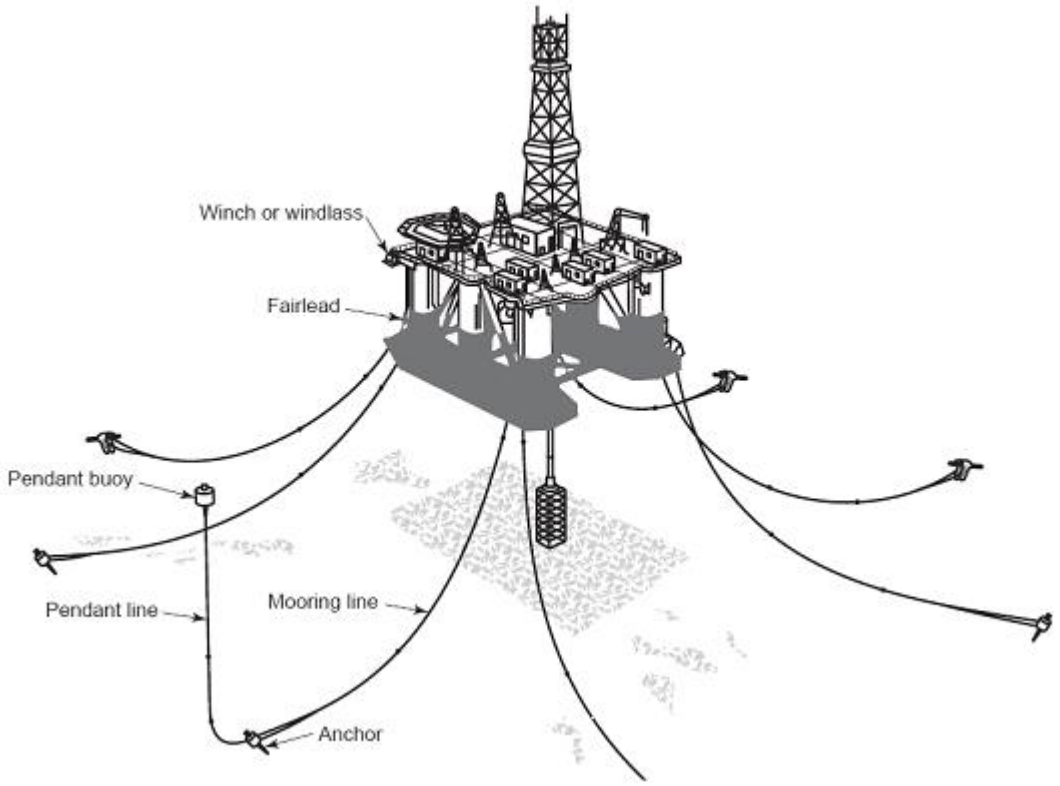

Figure 3.10 Spread mooring [49]

Different vessels may have different sensitivity to local weather with the spread mooring system. For example, a spread mooring system with a semi-submersible vessel or spar platform is relatively insensitive to the direction of environmental load, while FPSOs or ship shaped vessels are more sensitive to the environmental direction

[49]. Mooring lines can be chains, ropes, wires, or a combination of the three, and are detailed in section 3.3.1.

3.3.3.2 Single point mooring

The Catenary Anchor Leg Mooring (CALM), the Single Anchor Leg Mooring (SALM) and the Turret mooring system are the most popular single point mooring systems.

Figure 3.11 Turret mooring system [51]

Turret mooring, as the name indicates, has an internal or external turret attached to the vessel that allows it to rotate around anchor legs, while all the mooring lines are linked to the turret. Figure 3.11 shows an internal turret mooring system. Since first introduced in 1986, the turret mooring system has enabled vessels successfully weather vane into equilibrium headings under more extreme environmental conditions.

The design of turret in a vessel is complex task. The farther forward the turret from the mid-ship, the easier the vessel to cope with non-collinear environments. However, the vertical motions such as pitch of the vessel are increased with the turret placed further away.

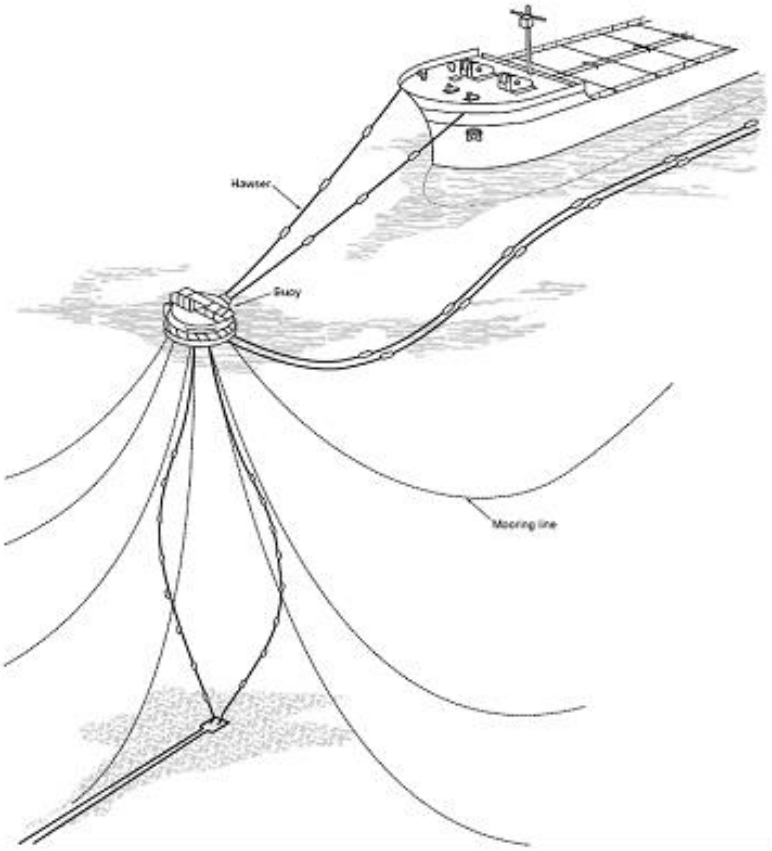

Figure 3.12 CALM system [49]

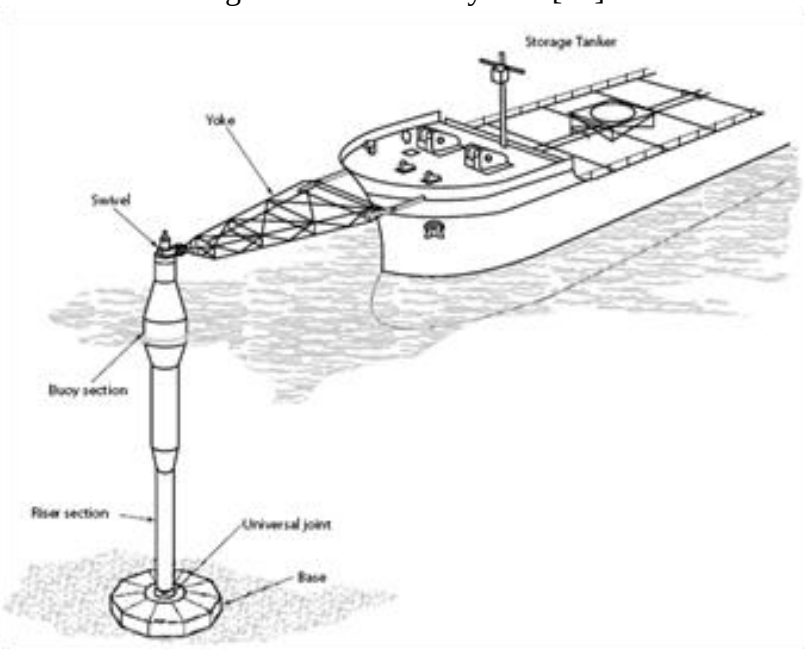

Figure 3.13 SALM system [49]

Figure 3.12 shows a CALM system which has a large buoy with catenary lines anchored to the seafloor. By using hawser(s) or rigid yokes with articulations, the vessel is connected to the buoy. Figure 3.13 illustrates a SALM system that contains a vertical riser section. The storage tanker is connected by flexible hawser(s) or soft yoke. Single point mooring systems are commonly used in relatively shallower depths of water [14].

3.4 OrcaFlex

OrcaFlex is one of the most prominent simulation commercial packages developed by Orcina. It is an offshore software for the design and analysis of a wide range of marine systems, including [53]:

- Riser systems: SCRs, TTRs, hybrids, flexibles, umbilicals, hoses.
- Mooring systems: spread, turret, SPM, jetty, etc.
- Installation planning with capabilities across the full range of scenarios.
- Towed systems: bundle dynamics, seismic arrays, towed bodies, etc.
- Defence, marine renewables, seabed stability, and many other types of systems.

One of its famous features that makes it stand out from all the other offshore packages is the powerful graphical user interface (GUI) introduced in 3.4.1.

3.4.1 OrcaFlex graphical user interface

OrcaFlex provides users with a very neat and powerful GUI to view the model in 3D and graphic dynamic results simultaneously. A typical OrcaFlex graphic interface is shown in Figure 3.14. It shows a single point spread mooring system with three mooring legs 120 °apart.

The GUI is consists of a main window that contains the menu, a tool bar, a status bar, and a sub-window that shows a 3D view of a model.

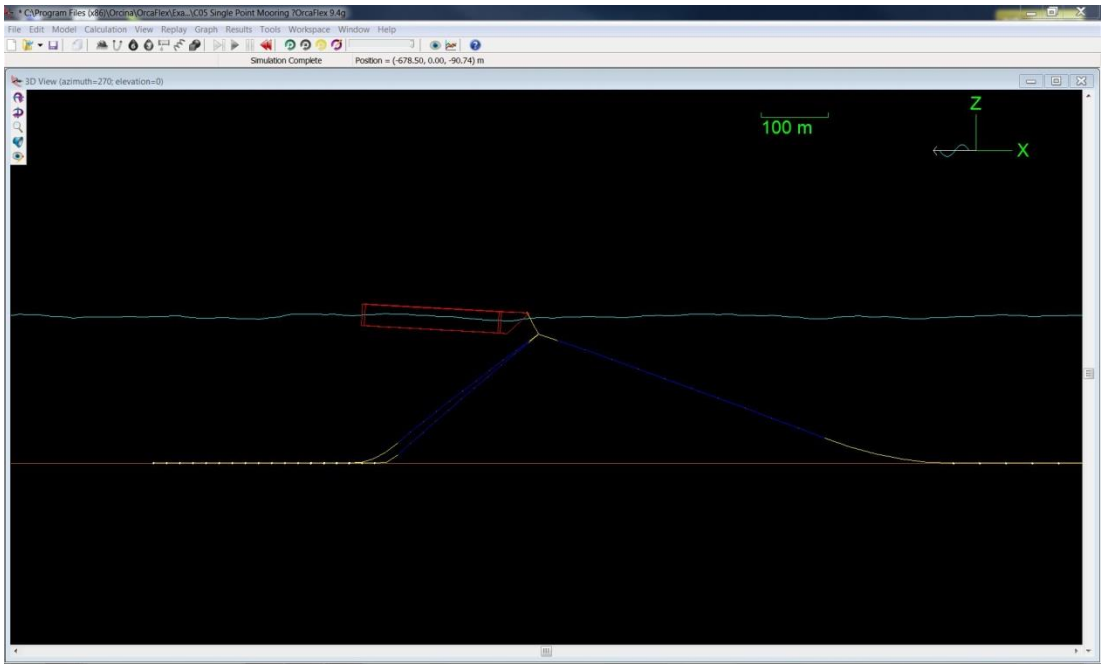

Figure 3.14 OrcaFlex example GUI

3.4.2 OrcaFlex model design & simulation

Each model has two standard objects, called general and environment. The general object contains general information, such as a description of the model, and the title, et cetera. The environmental information has environmental forces such as the current, wind, wave, and seabed.

Apart from the above two categories, there are particular objects that differ from model to model. The objects in those models may include the vessel, lines, 3D and/or 6D buoys, winches, links, and et cetera, depending on the complexity of the models. Details of all the objects are referred to in the OrcaFlex user manual [54].

3.4.3 OrcaFlex results overview

There are two alternative ways to extract results from an OrcaFlex simulation file, including:

1. OrcaFlex built-in results extraction GUI
2. Supplied Excel spread sheet for both bulk and single result post-process

The OrcaFlex built-in results extraction GUI is shown in Figure 3.15. The left hand section of the GUI is used to indicate the type of results one would like to extract.

The most frequently used type includes '*Summary Results*', '*Time History*', '*Range Graph*' and so on.

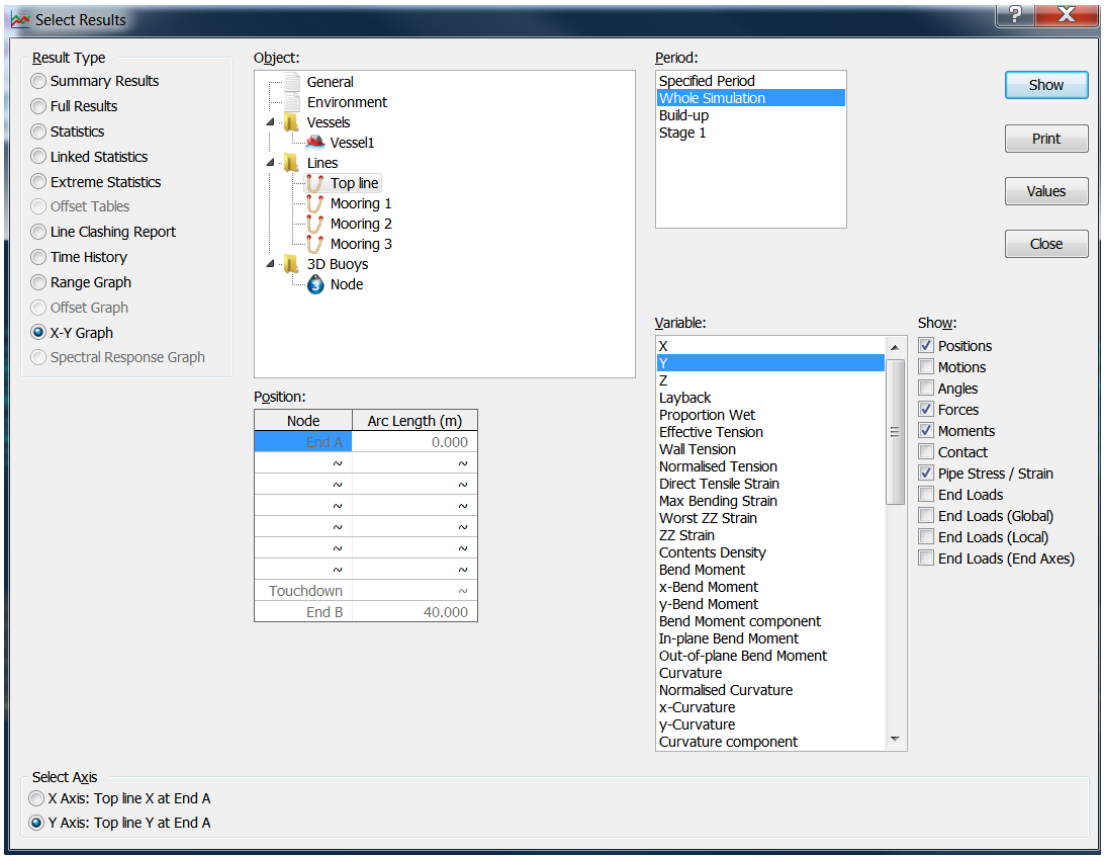

Figure 3.15 OrcaFlex built-in results extraction GUI

The second column counting from the left is called *Object* and that refers to objects in a simulation file. This section is used to choose which object in the model is of interest, to extract results. If lines are selected, the corresponding position of the lines should be specified in terms of nodes, otherwise the results of all the lines will show up. The furthest right column in this GUI is called *Period* and *Variables* in two sections. As OrcaFlex is a time-domain software, the period states the time sections along the whole simulation, that includes the option of '*Specified Period*', '*Latest Wave*', '*Wave Simulation*', '*Build-up*' and '*Stage 1*'. The variables section is used to choose the type of data the users are interested in, for instance '*End Moment*', '*Curvature*', '*Effective Tension*' and so on are the variables that can be chosen.

The typical time history graph shown in Figure 3.16 requires the time series data from simulation. Figure 3.16 shows that the following results have been extracted:

- Mooring that time history results is extracted: *Top line*
- Mooring position that time history results is extracted: *End A*
- Period that time history results is extracted: *Specified Period*
- Variables that time history results is extracted: *Effective Tension*
- Specified period that time history results is extracted: *from 0 to 500 seconds*

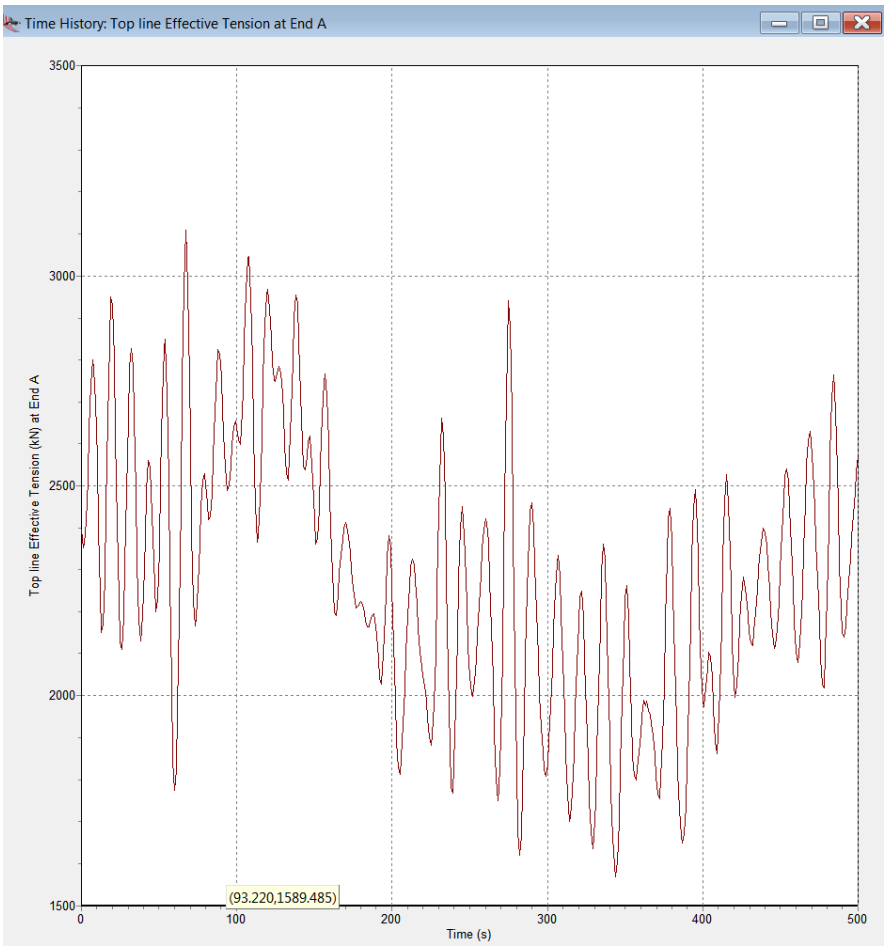

Figure 3.16 Time history graphical output

A typical range graph shown in Figure 3.17 extracts the following results:

- Mooring that range graph results is extracted: *Mooring 1*
- Mooring position that range graph results is extracted: *Entire Line*
- Period that range graph results is extracted: *Whole Simulation*
- Variables that range graph results is extracted: *Effective Tension*

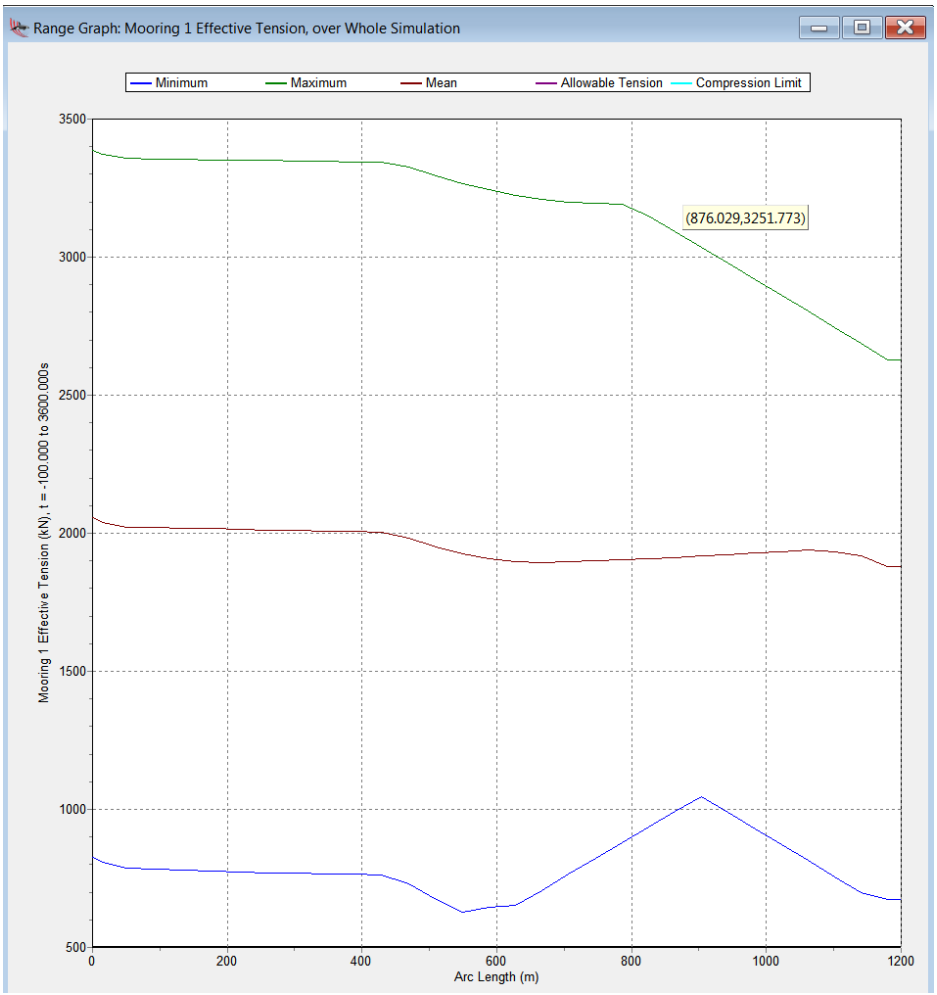

Figure 3.17 Range graph results output

3.5 Mooring line mechanics

A mooring system is conventionally made of steel linked chains and/or wire ropes that rely on an increase or decrease in line tension as they are lifted off or settled back down on the seabed. This change in tension produces a restoring force to balance the displacement of platform/vessel caused by the environment. According to Irvine [55], the non-linear stress strain relationship introduces a catenary shape to the cable hanging under its own weight. There are four functional requirements for the mooring system determined by the vessels, namely offset, lifetime, install ability, and positioning [42]. It is essential to understand the basic mechanics of a mooring line for the purpose of keeping the platform on station. Considering a typical profile of the catenary of a mooring line; it has a uniform vertical equilibrium in the Equation (3.1) with the isolated element shown in Figure 3.18

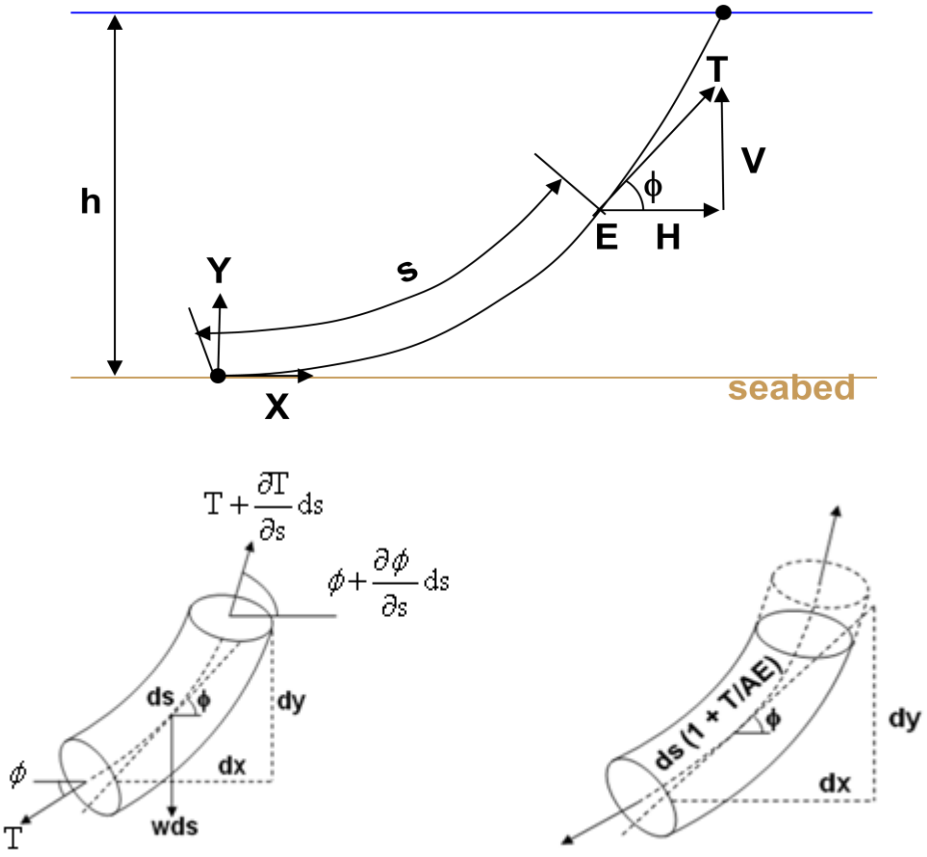

Figure 3.18 Isolated catenary elements with and without elasticity

$$\frac{d}{ds}\left(T\frac{dy}{ds}\right)=-mg=-w \tag{3.1}$$

where T is the tension in the cable, dy/ds is the sine value of the angle φ, mg and w are the unit self weight of the cable. Likewise, the horizontal equilibrium can be written as

$$\frac{d}{ds}\left(T\frac{dx}{ds}\right)=0 \tag{3.2}$$

Equations (3.1) and (3.2) are the force equilibrium for the catenary elements for both the vertical and horizontal directions. Solutions to those equations are provided with the negligence of bending stiffness of mooring material and direct external loading. The analytical solutions to those equations are obtained and summarised in Table 3.1 below.

Table 3.1 Static Results of Mooring Lines

	With elasticity	Without elasticity
Minimum line length (for a given tension)	$L = \frac{1}{w}\sqrt{T^2 - P_H{}^2}$	$L = L_Y\sqrt{\frac{2T}{wL_Y} - 1}$
Horizontal force (for a given tension)	$P_H = AE\sqrt{\left(\frac{T}{AE}+1\right)^2 - \frac{2wL_Y}{AE}} - AE$	$P_H = T - wL_Y$
Horizontal scope	$L_X = \frac{P_H}{w}\sinh^{-1}\left(\frac{wL}{P_H}\right) + \frac{P_H L}{AE}$	$L_X = \frac{P_H}{w}\sinh^{-1}\left(\frac{wL}{P_H}\right)$
Vertical force	$P_V = wL$	$P_V = wL$

3.6 Mooring flexibility iteration approach

The analytical solutions of mooring cables shown in Table 3.1 are not practically useful because all the solutions are inter-related to P_H which is the horizontal component of tension in the cable. However, P_H is not always available when calculating the profiles of the cable. Meanwhile, for mooring lines that are lying partially on the seabed, this analysis involves an iteration process to determine the touchdown point. In the algorithm, the line is progressively laid on the seabed segment by segment until the suspended part reaches equilibrium.

As mooring lines are subjected to large deformation due to their high flexibility, the behaviour of mooring lines is significantly different from stiff or solid structures. There are a number of approaches available to analyse this highly non-linear system, such as the energy based dynamic relaxation approach introduced by Lewis [56], and stiffness matrix method introduced by Krishna [57].

When the displacement of cable structures are not very large and the geometry of the system is well defined, it is common, even at the initial design phase to discretise the cable to bar-like elements and solve from numerical analysis and from algebraic equations [58, 59]. In terms of the geometric profile, however, the bar-like elements do not represent the real world and require a large number of elements.

The complete catenary geometry of a multi-component mooring line is determined by a procedure called flexibility iteration. This iterative approach was first suggested

by O'Brien [60, 61]. The approach presented herein is derived from the exact analytical solution based on O'Brien [61]. It is assumed that the stretching of the cables is purely elastic and axial, and the cable has no bending stiffness. Compared with other approaches such as bar elements, cables can be divided into fewer segments when subjected to a distributed load such as ocean currents. Hence, the current solution requires less computational effort and achieves fast convergence.

3.6.1 Derivation of Equations for the flexibility iteration

Consider the elastic cable element shown in Figure 3.19, which is naturally suspended under gravity in a vertical plane. According to Irvine [55], it has an equilibrium catenary profile under gravity load (self-weight) which satisfies Equations (3.1) and (3.2).

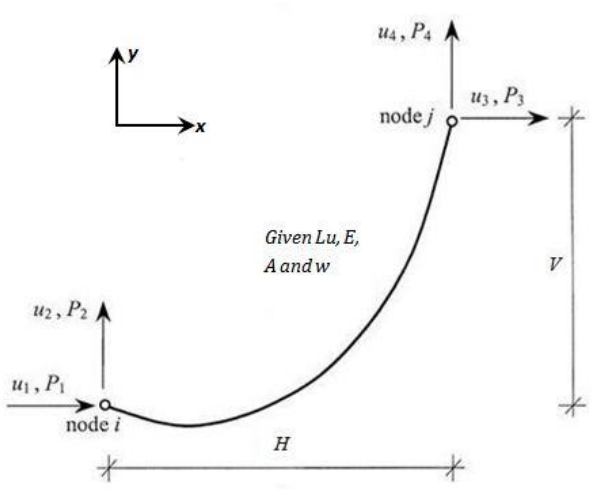

Figure 3.19 Catenary Cable Element

Integrating Equation (3.2) along the length of the cable (see Figure 3.18 bottom left) gives,

$$T\frac{dx}{ds} = P_H \tag{3.3}$$

where P_H is the horizontal component of cable tension which corresponds to P_1 and P_3 in Figure 3.19. Substituting identity $\frac{dy}{ds} = \frac{dy}{dx}\cdot\frac{dx}{ds}$ to Equation (3.1),

$$\frac{d}{ds}\left(T\cdot\frac{dy}{dx}\cdot\frac{dx}{ds}\right) = -w \tag{3.4}$$

Meanwhile, rearrange Equation (3.3) to the form of $\frac{dx}{ds} = \frac{P_H}{T}$, and then substitute to Equation (3.4), the classical differential equation of a cable subject to its own weight can be obtained as shown in Equation (3.5).

$$P_H \frac{d^2y}{dx^2} + w\frac{ds}{dx} = 0 \tag{3.5}$$

Due to the geometric constraint, which is $\left(\frac{dx}{ds}\right)^2 + \left(\frac{dy}{ds}\right)^2 = 1$, Equation (3.5) can be rewritten as

$$P_H \frac{d^2y}{dx^2} + w\left[1 + \left(\frac{dy}{dx}\right)^2\right]^{1/2} = 0 \tag{3.6}$$

After that, from the identity $1 + \sinh^2 t = \cosh^2 t$, let

$$\frac{dy}{dx} = \sinh t \tag{3.7}$$

to be substituted into Equation (3.6), therefore it can be further simplified as

$$P_H \frac{dt}{dx} + w = 0 \tag{3.8}$$

with some mathematical calculation.

Integrate Equation (3.8) in terms of *x* resulting in

$$t = -\frac{w}{P_H}x + \phi \tag{3.9}$$

Equation (3.10) can be integrated after substituting Equation (3.9) into (3.7).

$$y = -\frac{w}{P_H}\cosh\left(\frac{w}{P_H}x - \phi\right) + Constant \tag{3.10}$$

Given the boundary conditions of *x = 0, y = 0; x = H, y = V*, the constant of integration can be found as $Constant = \frac{P_H}{w}\cos\phi$, therefore

$$y = \frac{P_H}{w}\left[\cosh\phi - \cosh\left(\frac{wx}{P_H} - \phi\right)\right] \tag{3.11}$$

$$V = \frac{P_H}{w}\left[\cosh\phi - \cosh\left(\frac{wH}{P_H} - \phi\right)\right] \tag{3.12}$$

where $\phi = \sinh^{-1}\left[\frac{\lambda\left(\frac{H}{V}\right)}{\sinh\lambda}\right] + \lambda$ can be calculated by utilising the trigonometry identity $\cosh a - \cosh b = 2\sinh\frac{a+b}{2}\sinh\frac{a-b}{2}$.

To obtain the length L of the cable, one can take integration along x as

$$L = \int_0^H \frac{ds}{dx}\cdot dx = \int_0^H \left[1+\left(\frac{dy}{dx}\right)^2\right]^{1/2} dx = \int_0^H \cosh t \cdot dx \tag{3.13}$$

Substituting Equation (3.9) into Equation (3.13), the cable length L can be represented as

$$L = \frac{2H}{w}\sinh\lambda \times \cosh\left(\phi - \lambda\right) \tag{3.14}$$

Square Equation (3.14) and subtract from the square of Equation (3.12), it has the representation of $L^2 - V^2 = \frac{4H^2}{w^2}\sinh^2\lambda\cdot\cosh^2\left(\phi-\lambda\right) - \frac{4H^2}{w^2}\sinh^2\lambda\cdot\sinh^2\left(\phi-\lambda\right)$. After a few rearrangements, it can be simplified as

$$L^2 = V^2 + H^2\frac{\sinh^2\lambda}{\lambda^2} \tag{3.15}$$

where [62]

$$\lambda = \frac{w|H|}{2|P_1|} \tag{3.16}$$

$$P_2 = \frac{w}{2}\left[-V\frac{\cosh\lambda}{\sinh\lambda} + L\right] \tag{3.17}$$

Expand the item of $\frac{\sinh\lambda}{\lambda}$ in Equation (3.15) using a series expansion as $\left(\frac{\sinh\lambda}{\lambda}\right)^2 = \left(\frac{\lambda + \frac{\lambda^3}{6} + \ldots}{\lambda}\right)^2 \approx \left(1+\frac{\lambda^2}{6}\right)^2 = 1 + \frac{\lambda^2}{3} + \frac{\lambda^4}{36}$, and ignore the higher order terms, it can be represented as

$$1 + \frac{\lambda^2}{3} = \frac{L^2 - V^2}{H^2} \tag{3.18}$$

Further simplification of Equation (3.18) gives the coefficient λ as

$$\lambda = \left(6\sqrt{\frac{L_u^2 - V^2}{H^2}} - 6 \right)^{1/2} \tag{3.19}$$

3.6.2 The flexibility iteration formulation

As derived in section 3.6.1, the profile of a catenary cable relates vertical and horizontal projections, and the cable length *L* via Equation (3.15). According to Huang [63] and Chucheepsakul [64], the geometrical relationships integrated along the projections are

$$H = -P_1 \left(\frac{L_u}{EA} + \frac{1}{w} \ln \frac{P_4 + T_j}{T_i - P_2} \right) \tag{3.20}$$

$$V = \frac{1}{2EAw} \left(T_j^2 - T_i^2 \right) + \frac{T_j - T_i}{w} \tag{3.21}$$

where T_i and T_j are the cable tensions of the element at nodes *i* and *j* respectively, follows shown in Figure 3.19. *P* and *T* are related by Equations (3.22) to (3.25).

$$P_4 = wL_u - P_2 \tag{3.22}$$

$$P_3 = -P_1 \tag{3.23}$$

$$T_i = \sqrt{P_1^2 + P_2^2} \tag{3.24}$$

$$T_j = \sqrt{P_3^2 + P_4^2} \tag{3.25}$$

The expressions for horizontal and vertical projections *H* and *V* can be written for small changes in terms of P_1 and P_2 only by their first order differentials as

$$dH = \left(\frac{\partial H}{\partial P_1} \right) dP_1 + \left(\frac{\partial H}{\partial P_2} \right) dP_2 \tag{3.26}$$

$$dV = \left(\frac{\partial V}{\partial P_1} \right) dP_1 + \left(\frac{\partial V}{\partial P_2} \right) dP_2 \tag{3.27}$$

Rewriting Equation (3.26) and (3.27) in a matrix notation gives

$$\begin{Bmatrix} dH \\ dV \end{Bmatrix} = \begin{bmatrix} \dfrac{\partial H}{\partial P_1} & \dfrac{\partial H}{\partial P_2} \\ \dfrac{\partial V}{\partial P_1} & \dfrac{\partial V}{\partial P_2} \end{bmatrix} \begin{Bmatrix} dP_1 \\ dP_2 \end{Bmatrix} = F \begin{Bmatrix} dP_1 \\ dP_2 \end{Bmatrix} \tag{3.28}$$

where F is the incremental flexibility matrix and it is equal to the inverse of the stiffness matrix K:

$$K = F^{-1} = \begin{bmatrix} k_1 & k_2 \\ k_3 & k_4 \end{bmatrix} \tag{3.29}$$

When comparing Equations (3.26)~(3.29) with Equations (3.20) and (3.21), to ensure the matrix is invertible, it must have a non-zero determinant. Hence, Equations (3.30) to (3.32) are obtained by taking partial derivatives with respect to P_1 and P_2.

$$k_1 = -\frac{1}{\det F}\left(\frac{L_u}{EA} + \frac{1}{w}\left(\frac{P_4}{T_j} + \frac{P_2}{T_i}\right)\right) \tag{3.30}$$

$$k_2 = k_3 = -\frac{1}{\det F}\left(\frac{P_1}{w}\left(\frac{1}{T_j} - \frac{1}{T_i}\right)\right) \tag{3.31}$$

$$k_4 = \frac{1}{\det F}\left(\frac{H}{P_1} + \frac{1}{w}\left(\frac{P_4}{T_j} + \frac{P_2}{T_i}\right)\right) \tag{3.32}$$

where the determinant is given as

$$\det F = \left(-\frac{L_u}{EA} - \frac{1}{w}\left(\frac{P_4}{T_j} + \frac{P_2}{T_i}\right)\right) \times \left(\frac{H}{P_1} + \frac{1}{w}\left(\frac{P_4}{T_j} + \frac{P_2}{T_i}\right)\right) - \left(\frac{P_1}{w}\left(\frac{1}{T_j} - \frac{1}{T_i}\right)\right)^2 \tag{3.33}$$

3.6.2.1 Slack mooring cable

The idea of the flexible iteration method starts with an initial estimation of the horizontal and vertical projections H and V. Then, the differences between the actual projections and estimated projections are minimised until a tolerable error is found. In order to initialise the loop, reasonable estimations of P_1 and P_2 are required to ensure the convergence. The value for the horizontal component of the tension can be obtained from Equation (3.15) by substituting the stretched length L with the original cable length L_u. Keeping the first two terms of the series expansion of $(\sinh^2\lambda)/\lambda^2$, the λ value can be estimated via Equation (3.19).

By substituting Equation (3.19) into (3.16) and rearranging, an approximation of P_1 can be estimated. Likewise, substituting Equation (3.19) to (3.17), P_2 can be found directly. Karoumi [65] demonstrated that, with these initial values, convergence is achieved rapidly, generally within four to five iterations.

3.6.2.2 Taut mooring cable

If Equation (3.19) does not have a real root, this indicates a taut cable. That is a cable whose unstretched length is less than the distance between its ends. The initial position has a situation where L_u is shorter than the distance between nodes i and j. According to Peyrot [58], λ is about four times the sag to span ratio for horizontal span, a conservative estimate of sag to span ratio of five percent can be assumed. Therefore, an initial estimation value of 0.2 for λ can be applied in cases where the cable has a stretched and taut position. If the initial cable arrangement is vertical or near vertical, a large value of λ is applied (10^6) in order to stabilise the iterations.

Summarising the implementation process of the flexibility iteration method above, the initial components of tension force P_1 and P_2 are evaluated at the first stage. Then, cable projections H and V are obtained. The misclosure vector based on actual projections and the estimated projections $\{\Delta H, \Delta V\}^T$ can then be calculated. Corrections to the initial estimation of forces are available through computed misclosure vector as:

$$\begin{Bmatrix} \Delta P_1 \\ \Delta P_2 \end{Bmatrix} = K \begin{Bmatrix} \Delta H \\ \Delta V \end{Bmatrix} \tag{3.34}$$

$$\begin{Bmatrix} P_1 \\ P_2 \end{Bmatrix}^{i+1} = \begin{Bmatrix} P_1 \\ P_2 \end{Bmatrix}^{i} + \begin{Bmatrix} \Delta P_1 \\ \Delta P_2 \end{Bmatrix} \tag{3.35}$$

If the geometry of the whole cable is to be determined, coordinates for a number of points along the cable need to be computed. This process becomes very simple because both P_1 and P_2 are known after a few iterations. By substituting all the necessary values into Equation (3.20) and (3.21), the corresponding positions of each component can be calculated and therefore, the cable profile is obtained.

3.6.3 The flexibility iteration method for multi-component cable

Suspended cables subjected to self-weight can be determined efficiently by the approach introduced in sections 3.6.2. However, when the cable has multi-component constitutions and/or varying applied external distributed loads, the cable profile does not stick to simple catenary shape (self-weight only). The entire cable is then assembled from the individual stiffness matrices to form a system for which the equilibrium can be found by adopting the Newton-Raphson non-linear approach.

Since the cable is subdivided into components by nodes, the element tangent stiffness matrix K_t for the cable component can be obtained in terms of the four nodal degrees of freedom as ($k_2 = k_3$):

$$K_t = \begin{bmatrix} -k_1 & -k_2 & k_1 & k_2 \\ -k_3 & -k_4 & k_3 & k_4 \\ k_1 & k_2 & -k_1 & -k_2 \\ k_3 & k_4 & -k_3 & -k_4 \end{bmatrix} \tag{3.36}$$

Likewise, from Equation (3.34), the element tangent stiffness matrix K_t relates the incremental element force vector and the incremental displacement vector through Hooke's law

$$\begin{Bmatrix} \Delta P_1 \\ \Delta P_2 \\ \Delta P_3 \\ \Delta P_4 \end{Bmatrix} = K_t \begin{Bmatrix} \Delta u_1 \\ \Delta u_2 \\ \Delta u_3 \\ \Delta u_4 \end{Bmatrix} \tag{3.37}$$

A flowchart of the calculation process is shown in Figure 3.20. The tolerated error (TE) can assumed as an arbitrary small quantity, such as 10^{-5}. In Figure 3.20, each component of the cable is calculated through the flexibility iteration approach initially, and then the global tangent stiffness matrix of the structure is formed for the Newton iterations.

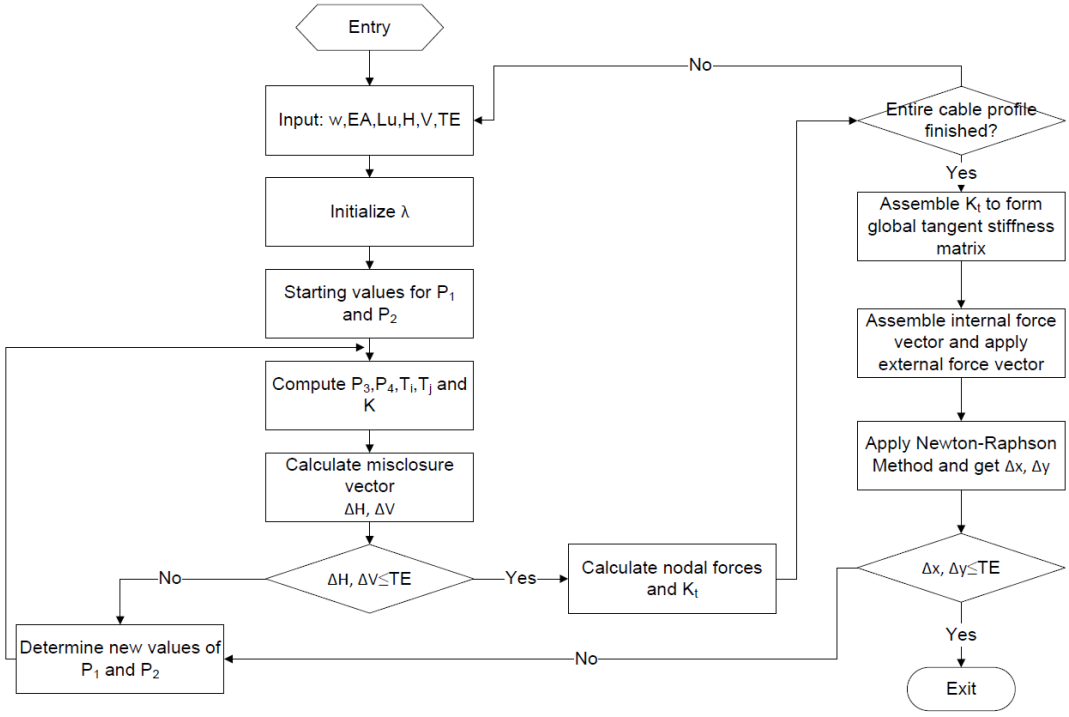

Figure 3.20 Numerical modelling flowchart of multi-component catenary cable

3.6.4 Improvement of the flexibility iteration method - Taut-slack algorithm search

The advantages of applying the flexibility iteration are the rapid converging speed and the natural catenary built component which resembles real behaviour [65]. However, this flexibility iteration approach does not always converge when looped in the Newton iteration. The reason for the divergence is because the flexibility iteration approach can only work in a smooth and continuous solution surface. When spikes or discontinuities occur, even a reasonably good initial estimation may still lead to instability in the solution space or a complete failure of the iteration. In that case, Andreu et al. [66] suggest using the bisection approach in the element resolution scheme for the sake of stability and accuracy. However, the bisection method converges linearly and the speed is slow, so it has not been adopted here.

For example, a multi-component cable has a taut component with an initial estimation value of 0.2 for λ as suggested by Peyrot [58]. This taut component means that the unstressed length L_u is shorter than the distance between node i and j which are the terminal points. The flexibility iteration approach searches for the equilibrium based on the initial estimation of λ in the taut zone until the equilibrium position is

found. However, it is possible that the stressed cable under its self-weight has been elongated due to elasticity when hanging in a working condition. Even with a small elongation, the cable may switch from taut to slack.

Another possibility for causing divergence also occurs in looping the multi-component cable with the Newton-Raphson method. Assuming one has a slack component with an estimation value of λ based on Equation (3.19). Since the segments' end positions (intermediate nodes location) of the cable keep changing in the Newton iterations, it is highly likely that at an intermediate step the cable component can become taut. The flexibility iteration approach, nevertheless, keeps searching for equilibrium in the slack range, which results in a divergence of the approach.

As mentioned above, the flexibility iteration approach cannot always guarantee a convergence when applied to multi-component cables. To improve the stability, it needs an algorithm to smooth the calculation process from taut to slack and vice versa. At the occurrence of divergence, a switch has been placed in the calculation. The switch starts to take action when it detects instability. It terminates the on-going calculation and then assigns a new initial estimation that is always in the opposite range of the previous, to re-run the simulation.

For instance, when an initial trial set of tension force components, obtained from the slack condition, fails to converge, the switch (taut-slack algorithm) terminates its calculation, and re-assigns a new trial set of values from the taut condition. The application of this switch ensures a fast convergence of the flexibility iteration approach. This works well even if in an inferior value of λ is chosen initially. This switch of initial conditions in the calculation is the 'taut-slack' algorithm. An example demonstrating the application of the 'taut-slack' algorithm is outlined in the section 3.6.5.

3.6.5 Examples of the taut-slack algorithm search

The main application of the taut-slack algorithm is in the numerical solution process of multi-component cables using Newton-Raphson method. The divergence always

occurs in the vicinity of the boundary between taut and slack during flexibility iteration. It is rare to see this occurring when using the flexibility iteration for a single component cable. However, it is common while in Newton-Raphson numerical iteration where the model contains multiple cable components, and those components have been frequently changing positions during iterations.

3.6.5.1 An example of cable segment

A cable of uniform properties has an unstressed length of 200m and axial stiffness $EA = 1.3 \times 10^9$ N, hanging in a vertical plane with horizontal and vertical projections of 150m and 100m respectively. The unit weight of the cable is 664.4 N/m. The Newton-Raphson method incorporated with flexibility iteration approach has been used in this example. The total cable was divided into six segments with equal length.

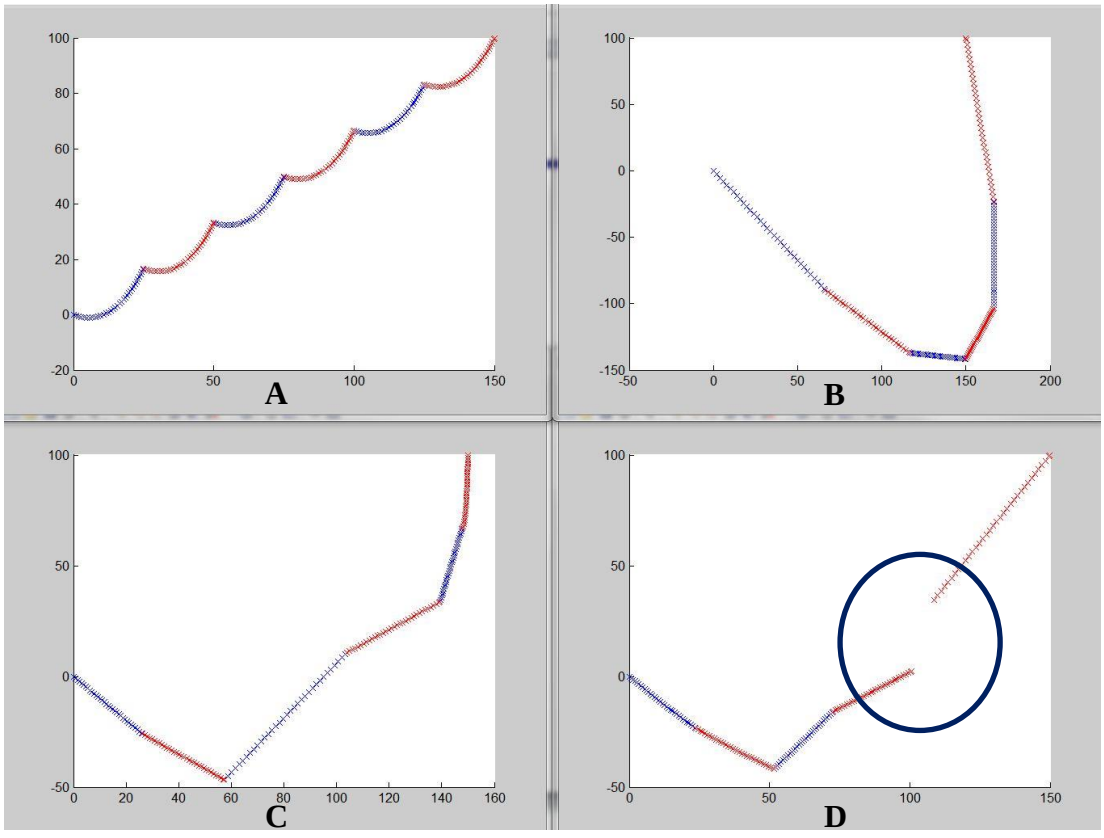

Figure 3.21 Steps of Newton-Raphson iteration of a mooring cable

The initial positions of these six segments cable can be assumed in a straight line connecting the starting and ending location of this cable. For each segment, the

flexibility iteration approach initially gave a natural catenary shape, as shown in Figure 3.21A. At the second step of triggering the Newton-Raphson method, shown in Figure 3.21B, more than one of the segments was stretched due the nodes changing position. Likewise, the third step still shows the process of a search for equilibrium. The intermediate steps while applying the Newton-Raphson method caused the uncertainties of the cable segments being slack or stretched.

As highlighted in Figure 3.21D, which was the fourth step of the Newton-Raphson method in this particular example, one of the segments cannot be plotted due to its failure to converge so the global stiffness matrix cannot be assembled because this part is absent. Therefore, no results can be drawn with this cable. In other words, the Newton-Raphson method cannot fulfil the task as the failure of the flexibility iteration approach of a segment. In order to have a better understanding of the problem, it is worthwhile examining the failure segment.

3.6.5.2 Application of the taut-slack algorithm

By looking into the failure segment, which is one component of a mooring cable with a total length of 200 metres, the length of the component is one-sixth of the total length, which is 33.3333m. Details of the failed segment have been extracted from the whole cable, as shown in Figure 3.22. The section 3.6.5.1 shows that running the original flexibility iteration alone with the Newton-Raphson method would result in a divergence of the calculation and no solution to the question. This section examines the failed segment shown in Figure 3.21D with the taut-slack algorithm. Since the segment is still part of the whole cable, the properties remain unchanged for the following analysis. The horizontal and vertical projections of this segment are based on an examination of the third step of running the Newton-Raphson iterations, so the results to 4 decimal places have been kept as an illustration.

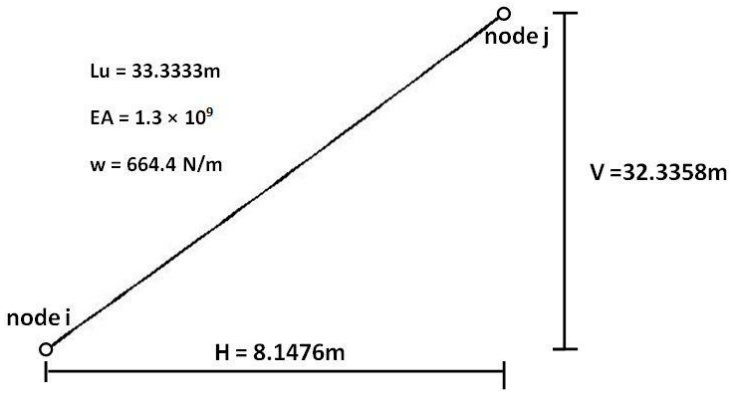

Figure 3.22 A Segment of a Catenary Cable

Figure 3.22 depicts the failed segment from the cable shown in Figure 3.21D, with an unstressed length of 33.3333m and an axial stiffness $EA = 1.3 \times 10^9$ N. The horizontal and vertical projections of this component are 8.1476m and 32.3358m respectively.

Analysing this particular segment with the flexibility iteration approach alone, the distance between node *i* and *j* is $\sqrt{8.1476^2 + 32.3358^2} = 33.3464$ m > 33.333 m, therefore, this segment was stretched. An initial estimation of λ equal to 0.2 has been considered for the iterations, but the flexibility iteration does not converge with this λ value and the results from using this approach are not available. When calculating the real length of this segment, which is 33.3466m, it is still longer than the distance between node *i* and *j*. In other words, this segment is still in a slack condition which in reality proves that the initial estimation was incorrect, and therefore divergence was detected during iteration.

Table 3.2 Results comparison of tension

	With 'Taut-slack' algorithm	**Without 'Taut-slack' algorithm**	**OrcaFlex**	**Differences (%)**
Top tension (kN)	530.42	Divergence/no solution	530.52	0.02
End tension (kN)	508.94	Divergence/no solution	508.94	0

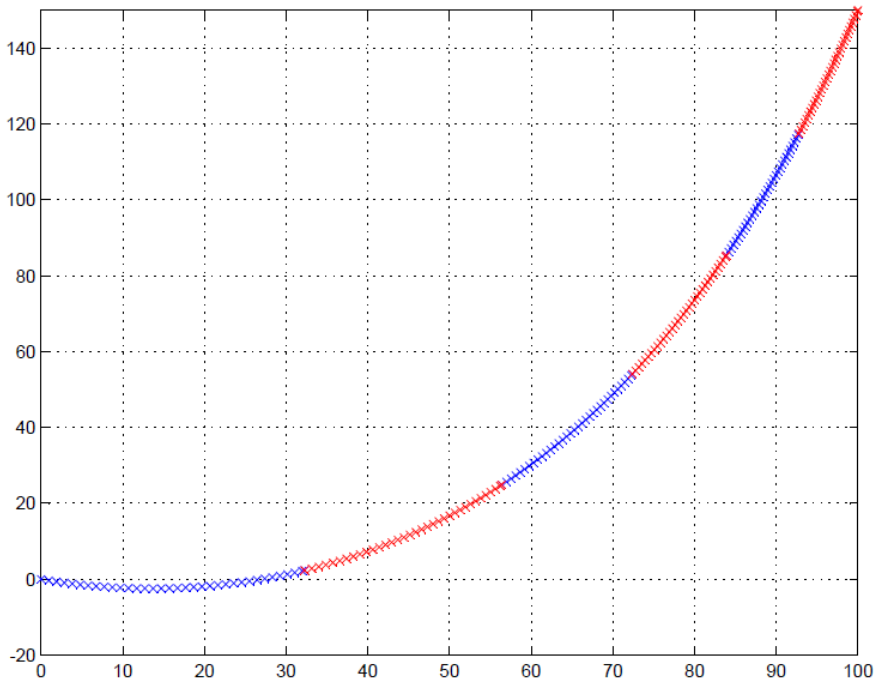

Figure 3.23 Final profile of the example cable

When the taut-slack algorithm detects the divergence, it re-assigns a value of 0.05 to λ, but with the flexibility iteration approach, the λ value claims that the cable component is in a slack state instead of being taut. It can be seen in Figure 3.22 that all the lengths have retained four significant figures after the decimal point, so if a simulation is carried out without the taut-slack algorithm, the overall response of the cable is failure due to divergence in the second segment counted from the top. However, the profile of the segment can be found with help from the taut-slack algorithm.

As a comparison, the total profile of the entire cable can be plotted (shown in Figure 3.23) with the converging solution obtained from the taut-slack algorithm. The results of the second segment counted from the top, as shown in Figure 3.21D, are summarised in Table 3.2.This example has been incorporated in MATLAB code and the results were compared with simulation from OrcaFlex [54]. The profile of the entire cable simulated from OrcaFlex is shown in Figure 3.24.

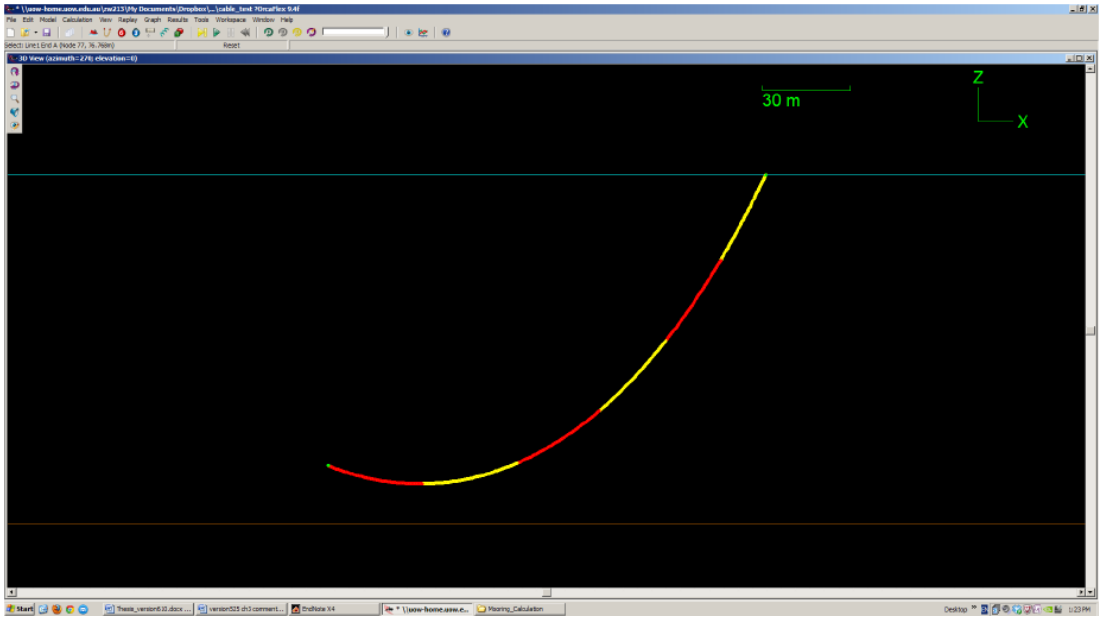

Figure 3.24 Final profile of the example cable from OrcaFlex

It is clear that in the example shown here, the taut-slack algorithm improves the stability of the flexibility iteration approach for convergence, while also retaining the advantages of the flexibility iteration approach, such as fast convergence and good accuracy. With the taut-slack algorithm, multi-component cable simulations can easily be accomplished in the Newton-Raphson iterations without a significant increase in the cost of computation.

3.7 Mooring system analysis

The mooring system consisting of mooring lines placed in a certain pattern to keep the vessel in a limited position is known as the station keeping system. In a mooring station keeping system, the system is analysed as a whole. A schematic view of a mooring pattern is shown in Figure 3.25 below.

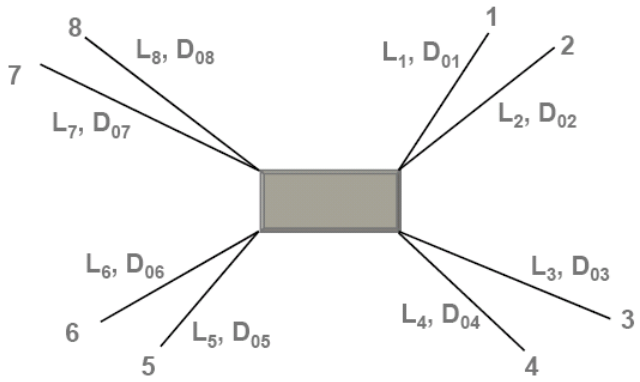

Figure 3.25 Schematic diagram of a mooring system

In the analysis, a preliminary understanding of the force of a mooring line is that it depends on the length and submerged weight of the line, the depth of water below the fairlead used to guide the mooring cables, and the horizontal distance from the anchor to the fairlead. The force of each catenary mooring line has been calculated by giving coordinates to the end point of the line while including its length and elasticity. In a spread mooring system, these forces are then added together to give the total horizontal and vertical restoring forces. The line under the biggest load is then analysed by placing the vessel in a prescribed manner at every direction away from its initial position. It is necessary to find equilibrium to the system as a whole, instead of a single mooring line. Hence, the equation as a system is [51].

$$\begin{cases} \sum_i T_{Hi}(r_i)\cos\theta_i = 0 \\ \sum_i T_{Hi}(r_i)\sin\theta_i = 0 \\ \sum_i X_i T_{Hi}(r_i)\sin\theta_i - \sum_i Y_i T_{Hi}(r_i)\cos\theta_i = 0 \end{cases} \tag{3.38}$$

where T_{Hi} is the initial horizontal tension for a mooring line,

θ_i is the initial mooring line horizontal angle to the X direction,

r_i is the horizontal distance from anchor to fairlead determined by the unknown vessel displacements Δx, Δy and $\Delta\theta$ defining the vessel position,

X_i, Y_i are the coordinates of fairleads in global axis system.

X_i, Y_i are the coordinates of fairleads in global axis system.

For mooring dynamic analysis, there are several available methods. There are linear spring, non-linear quasi-static (QS) catenaries subject to mooring forces and their buoyancy including hydrodynamic loading and fully dynamics. The aim of the dynamic analysis is to analyse the mooring system as part of the response of a moored vessel. As a simple model established in Figure 3.26, following general equation of motion need to be solved.

$$M\frac{d^2\vec{x}}{dt^2}+B\frac{d\vec{x}}{dt}+C\vec{x}=\vec{F}_{static}+\vec{F}_{WF}+\vec{F}_{slowdrift}+\vec{F}_{mooring} \tag{3.39}$$

where *M*, *B*, and *C* are the matrices of hydrodynamic mass, damping and stiffness respectively, and *x* is the displacement vector. *F* shown in Figure 3.26 represents the total sum of the forces in equation (3.39).

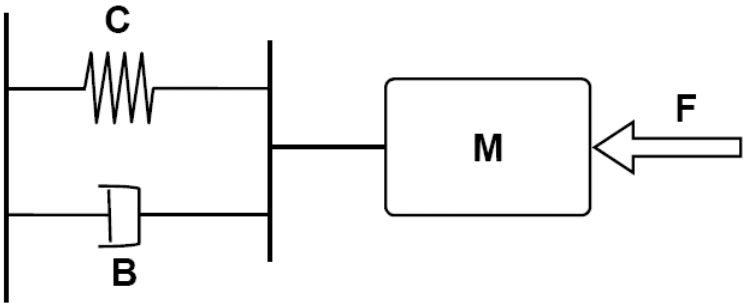

Figure 3.26 Simple model of mooring analysis

Of all the methods mentioned above, linear spring analysis is of the simplest because all the non-linear characteristics are replaced by linear components in the calculation, so it can be solved by the frequency domain. However, as the complexity of non-linearity is simplified by the linear function, the expected solution is not as accurate as with other methods. Other methods are time domain, which means the fully dynamic time domain analysis can give a dedicated solution which could be used in the final design process.

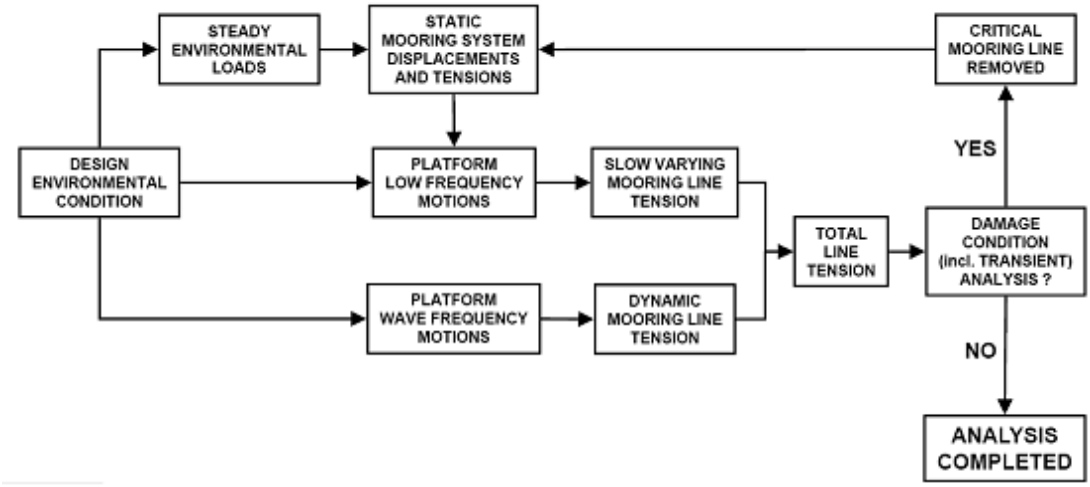

Figure 3.27 Mooring analysis flowchart [51]

Considering the external environment is the first step in the conceptual design of the system. Two classifications should be taken into account. One is the maximum extreme design situation that contains a combination of wind, wave, and currents, which cause an extreme load. Different design criteria may be used here according to different codes and/or environments. The other is the maximum operating situation that includes a combination of wind, wave, and current in which the unit can continue to work. However, the maximum operating situation should never exceed the maximum design condition. According to Mombaerts [51], a mooring analysis flowchart can be seen in Figure 3.27.

The mooring system has to be analysed and simulated at the damage situation where the most loaded mooring was assumed to be broken. if a line breakage happens, the vessel would oscillates in a new mean position and the second most loaded line should be analysed in damage condition again.

3.8 Mooring system design

This section discusses mooring design procedures and some uncertainties associated with their inputs. There are three basic methods: static design, quasi-static design, and dynamic design respectively.

3.8.1 Static mooring design

A static design is often used at the initial stage for conceptual design purposes. The system is analysed through load and excursion characteristics for the most loaded mooring line by ignoring the fluid dynamic forces. This analysis is carried out using the catenary equation discussed in section 3.5. After placing the vessel through prescribed distances (normally only surge and sway motions are considered), a diagram of vessel excursion versus the steady component of environmental forces can be depicted, as shown in Figure 3.28. The slope of the curve represents the stiffness.

One of the obvious disadvantages of this method is that the dynamic components are not included, so due to the absence of dynamic features, conservative assumptions must be made to account for uncertainties that may be involved in the design afterwards. However, it is helpful to carry out this type of design at the initial stage.

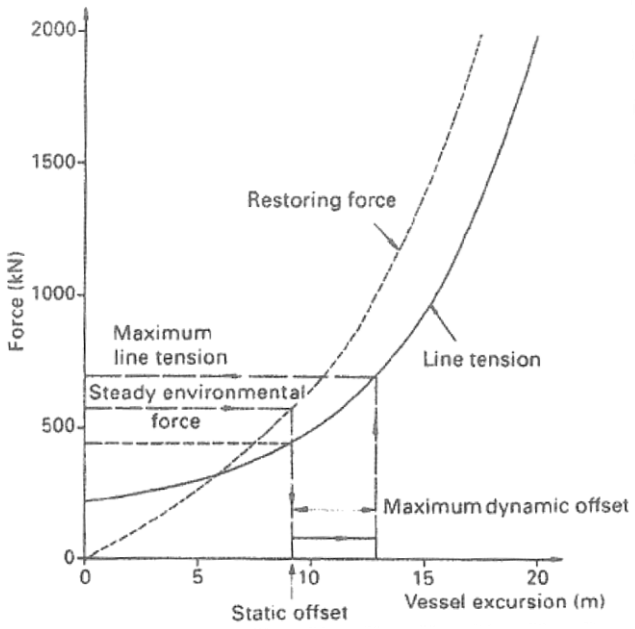

Figure 3.28 The result curve of a static design for the most loaded line [42]

3.8.2 Quasi-static (QS) mooring design

As with the static design, environmental forces such as wind and current are still taken as a steady component in this method. The difference is that the wave induced

force for a time-domain simulation is allowed in QS design. Therefore, the equation of motion (Equation (3.39)) has been used with some damping contributions from the vessel. This design procedure is more complex than static design because to give a relatively reliable answer, simulating a QS design must cover a minimum of 18 hours of full scale behaviour to provide enough low frequency offsets.

3.8.3 Dynamic mooring design

A dynamic design is based on the static design which provides the equilibrium and static configuration of the mooring system. A fully dynamic simulation uses lumped mass finite element schemes to model the mooring lines as small segments in order to catch the dynamic features of the system. Figure 3.29 shows a diagram of a design study in a time domain manner.

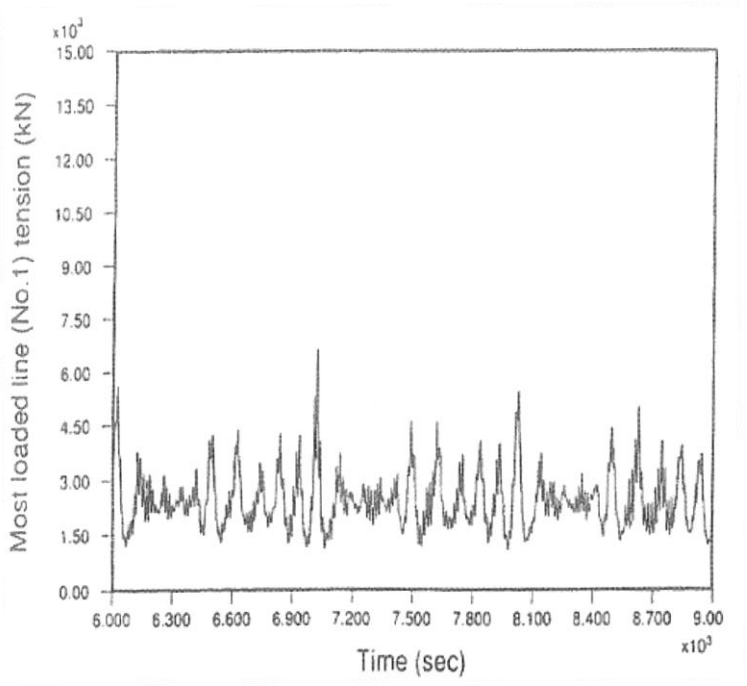

Figure 3.29 Example of mooring line tension in dynamic analysis [42]

In order to have accurate results, the time steps must be small to include the wave induced mooring oscillations. Therefore, this method is computationally expensive in terms of simulation time and different load cases must be carried out for the vessel in multi-directional weather conditions.

3.8.4 Hybrid mooring design & analysis

A dynamic analysis of the mooring system is complex due to the action of the coupling between the platform and the risers. The purpose of designing a station keeping system is to serve the risers operating within limited offsets.

In the typical design of an offshore system shown in Figure 3.30, information regarding the vessel is given by the clients or ship contractors at an early stage. The type vessel must be decided at the very beginning because the subsequent steps of design and analysis all depend on the vessel. After the vessel has been selected, the next step is the design of the station keeping system (mooring). However, from a practical engineering perspective, the mooring requirement (allowable vessel offset) is often given by riser engineers so that risers can be in service without being challenged too much.

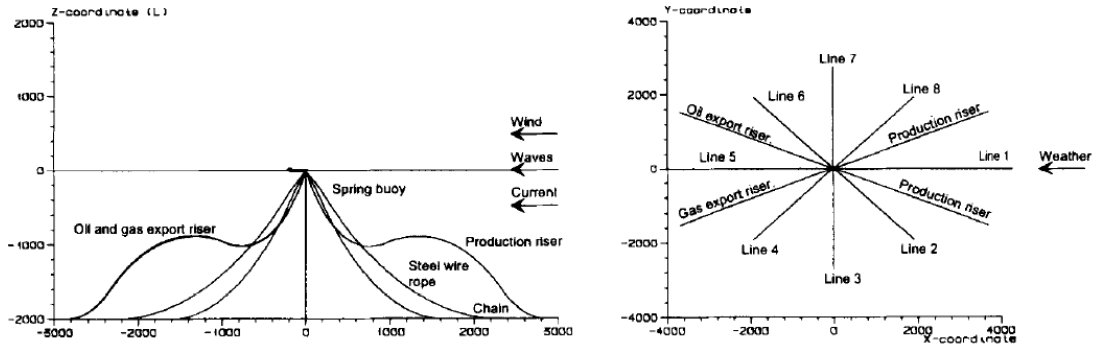

Figure 3.30 Schematic of station keep system and riser configuration [67]

The traditional method of assessment adopted by industry, and which is also utilised by some independent verification agencies (IVA), is to apply platform motions resulted from environmental simulation to the top of riser system [68]. This is also called de-coupled methodology. Rodrigues et al [69] summarised this traditional methodology that is utilised by the industry in the following steps:

- A characteristic offset, include both static and low frequency (LF), is applied to the top of the risers when the vessel and the moorings are considered.
- Wave frequency (WF) motions applied to the top of the risers are obtained when the Response Amplitude Operators (RAOs) are obtained without considering the lines.

- The risers are then modelled in detail and simulated in a time domain where all the representative offsets and platform WF motions as forced boundary displacements are considered.
- Verify the system according to the main industry standards, such as the criteria presented in DNV-OS-C201[70] or API-RP-2SK [13].

Currently, a few research projects have been conducted on coupled analysis, and many articles have been published on various technologies. Chen and Chai [71] conducted a time domain analysis on parametric studies of taut-wire mooring and semi-submersible platforms. Low and Langley [72, 73] simulated the time domain to acquire the low frequency motion, and frequency domain to acquire the wave frequency motion to minimise the efforts needed to couple the whole offshore system. In order to shorten the simulation time, Guarize, et al. [74] trained an Artificial Neural Network (ANN) to predict long response time histories for the slender rises of coupled vessels and mooring analysis.

3.9 Mooring design procedures in an engineering practice

This section introduces a practical way of designing a mooring system from scratch. Offshore oil production and exploitation begins with a flow assurance, which is a term coined by Petrobras in the early 1990s. It is a diverse discipline that involves many discrete and specialised subjects such as modelling, transient, geomechanics, handling solid deposits, and et cetera. It literally means a guarantee of flow.

All the information gathered during the flow assurance study is then posed and analysed by production designers such as riser engineers. Due to its initial and preliminary position, any mishap with flow assurance can lead to astronomical financial loss and catastrophic disaster, so the results of flow assurance cover the quality and quantity of stored hydrocarbon, difficulties with exploitation such as extreme temperature and pressure, and pre-designed well locations, and et cetera. When combined with a floating production system or platform detail, the risers can be then designed. Constraints such as vessel offset to keep risers production are passed to mooring design engineers to design the mooring system. A flowchart of this mooring pre-design phase is shown in Figure 3.31.

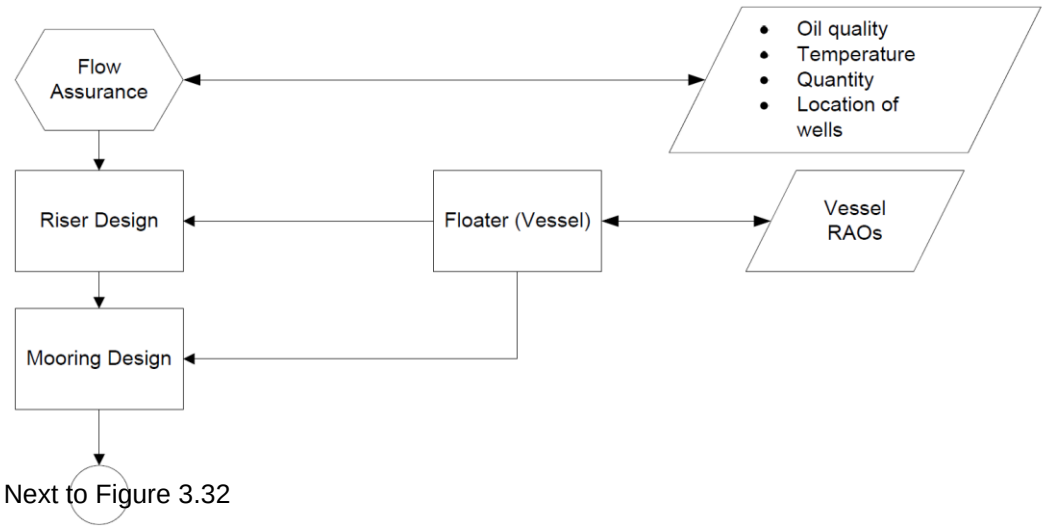

Figure 3.31 Mooring pre-design phase

From consultations with AMOG offshore engineers, there are five components involved in designing a mooring system: selecting the mooring pattern, configuration design, sizes and length, and pre-tension, respectively. Referring to section 3.3.3, mooring systems are categorised into two groups, namely single point mooring (SPM) and spread mooring system (SMS). SPM are more sensitive to environmental load directions while SMS are more suitable for semi-submersible types of platforms or vessels.

There are a number of criteria that must be considered when designing a mooring pattern, as shown in Figure 3.32. First of all, the weather conditions where the vessels need to be moored must be taken into account. For example, extreme weather such as storms approaching from a certain direction, in which circumstance it would be beneficial to design the mooring system to balance the forces. Second, undersea spaces for installing mooring lines need to be checked because any limitations mean that the positions of the lines and anchors may need to be designed in a particular pattern. Third, issues such as the possibility of mooring lines clashing with risers must be avoided, and finally, the spatial limitation in an SPM, such as the turret, may also become a constraint if the lines cluster together.

Mooring line configurations is heavily related to the material used to manufacture the mooring legs. In shallow waters a vessel or platform’s payload is not affected very

much by the mass of the mooring system, mooring legs made of steel chain links are preferred. The behaviour of steel chain mooring line can best be described by the catenary equation covered in section 3.5.

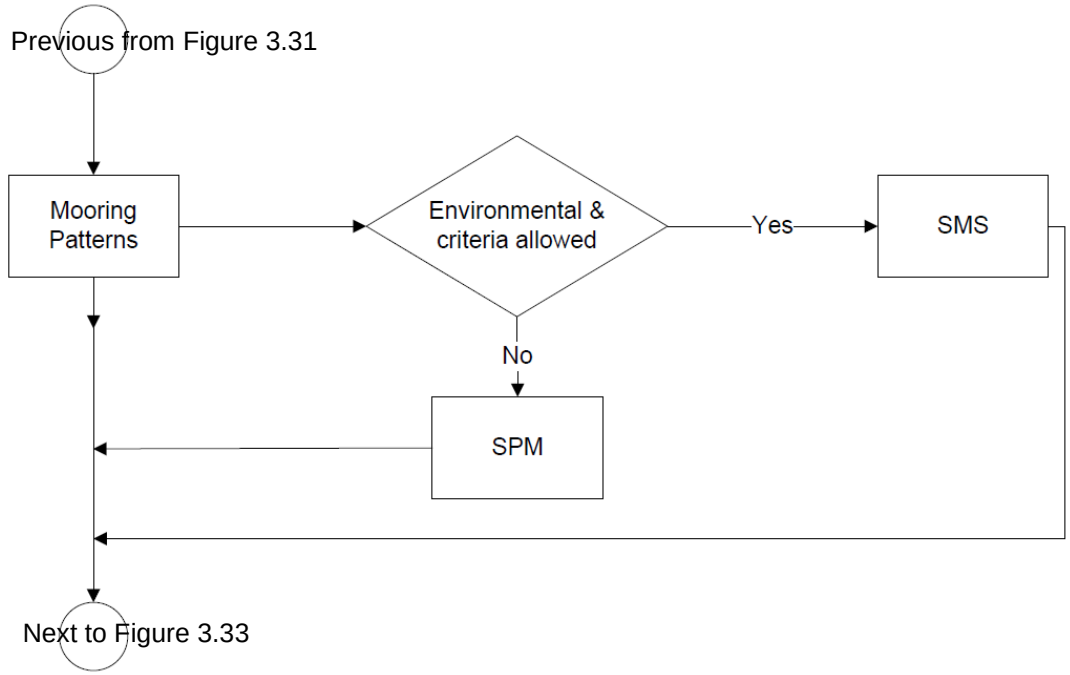

Figure 3.32 Selecting the mooring pattern

As the depth of water increases, the self weight of the mooring system can consume a large proportion of the vessel's payload. In deep water applications, steel catenary mooring chain links are too heavy for floating platforms so synthetic fibre would be a better option because they are much lighter than steel catenary chain links. Apart from their light weight, synthetic fibre lines can be flexible and absorb imposed dynamic motions. It also can reduce the length of mooring lines and parts having to be laid on the sea bed.

The inherent complexity in its mechanical properties makes synthetic lines more difficult to analyse than traditional ropes so synthetic mooring designs tend to be conservative in terms of their safety factors. Sometimes, multi-segmented mooring legs are preferred in terms of their self-weight. Mooring configuration can vary from 'chain – heavy chain – chain' to 'chain – fibre/steel wire – chain' type for the three

segments of the mooring lines. As shown in Figure 3.33, the configuration of mooring lines is directly related to the depth of water of the project. In relatively shallow water, single steel catenary chain links or ‘chain – heavy chain – chain’ type can be used, but as the depth of water increases, multi-component ‘chain – fibre/steel wire – chain’ type mooring lines are preferred. When a vessel has to be moored in deep water, synthetic taut mooring may be selected.

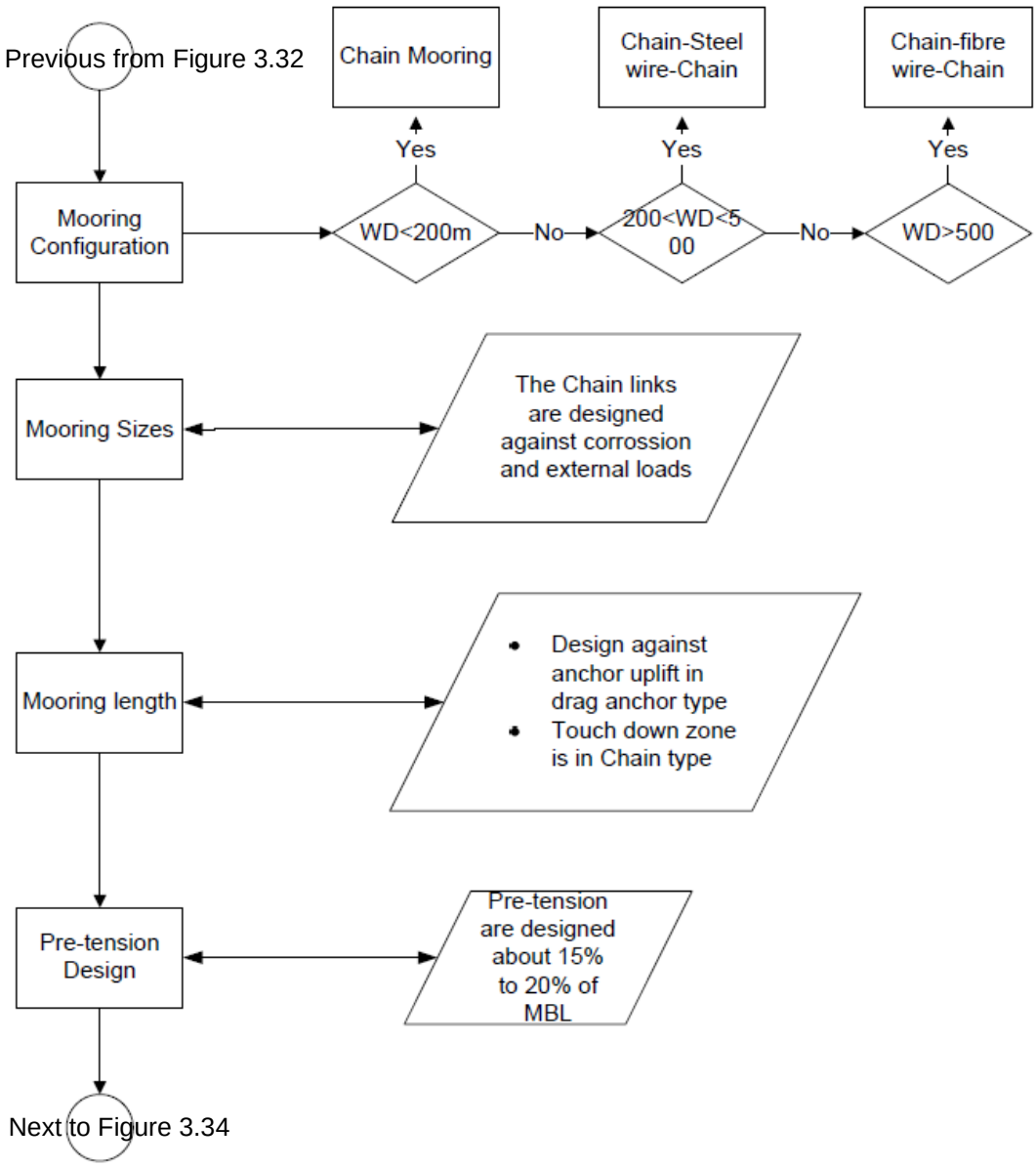

Figure 3.33 Mooring line designs

The sizes of the links of steel catenary chain link mooring lines are designed to cope with the steel corroding, and their breaking force. As the length of a single mooring

leg is normally hundreds of metres long, and sometimes thousands of metres, even a 1mm increase in the diameters of the chain links, can seriously affect the system both financially and in terms of its payload. As discussed in section 3.3.1, steel linked chain is categorised as either stud-link or studless chain. Studless chain is preferred for deep water permanent moorings because removing the stud can reduce the unit weight of chain and increase its fatigue life. However, removing the stud in the chain link increases the difficulty of handling the mooring lines. In other words, scarifying the convenience of easy handling of the mooring chain lengthens the design life of the mooring system.

There are three general concerns when designing the length of a steel mooring leg. The first concerns the type of anchor. For example, if a drag anchor used in the system it is important to have an uplift force that is less than the limitation. Controlling the uplift force can be done by changing the length of the mooring lines. The second concern is the restoring force provided by the system must be able to hold the system within the offset of a predetermined position, and the third issue is to always make sure that the bottom segment of chain is long enough to cover the whole touch-down zone.

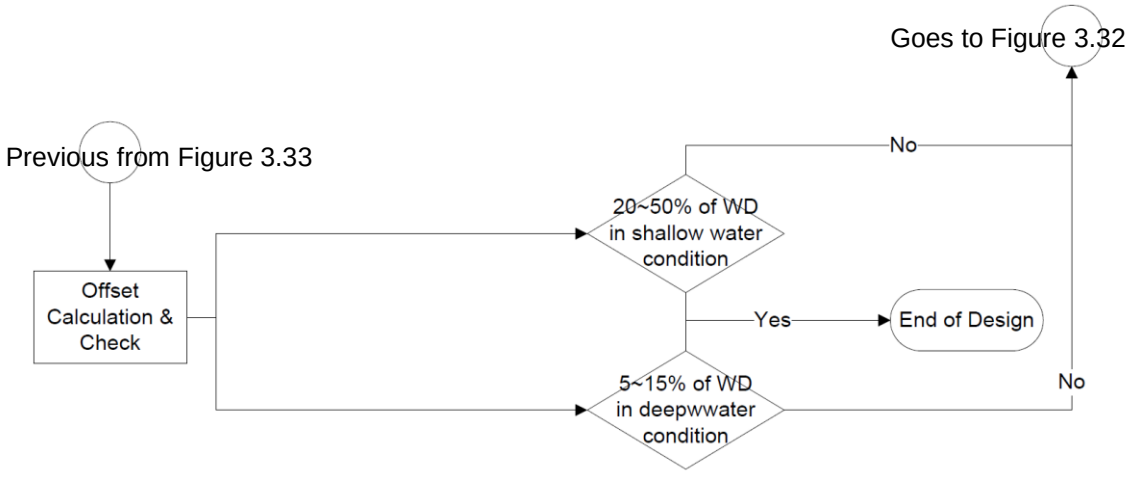

Figure 3.34 Checking the mooring offset check

One of the most important checking criteria for station keeping is the allowable offset of the vessel under both working and damaged conditions. Based on engineering experience, and as a rule of thumb, the offset for shallow water may vary from 20 to 50 per cent of the water depth, while in deep water the offset has been

limited to 5 to 15 per cent. In damaged cases, the mooring system should be able to survive extreme environmental states to minimise risks. As shown in Figure 3.34, if the offsets check has been satisfied, the mooring design can then be stopped, but if the offset requirement is not being met, the mooring lines must be re-designed. Variables that could be changed to make the mooring system work include the length of the mooring line, including its length, size, and levels of pre-tension, as shown in Figure 3.33. However, clients often choose their preferred mooring pattern so it is not advisable to modify the patter without careful planning.

3.10 Offshore mooring codes and standards

There are a number of recognised design codes and standards that cover offshore mooring design and analysis. In the reality of designing a mooring system it is necessary to draw upon more than one source, but this does not mean that in a single project, the input data, methods of analysis and safety factors can be combined from different sources. One should always be alarmed that combining the least conservative methods from different codes or practices is not allowed in the design.

One of the most modern and widely accepted standards for offshore mooring systems is the International Standard ISO 19901-7 [50], *Station keeping systems for floating offshore structures and mobile offshore units*. This document specifies methodologies for the design, analysis, and evaluation of floating structures, and the assessment of station keeping systems. The International Standard ISO 19901-7 [50] cannot be applied to vertical moorings or TLPs because only covers spread mooring systems and single moorings to the extent to which the requirements are relevant. This standard requires that a mooring system to be analysed for intact, redundant, and transient conditions. The redundancy condition is where the vessel has a new mean position after a single mooring line has broken, and the transient condition is where a vessel transients between being intact and redundant. A brief summary of the analytical methods has been tabulated in Table 3.3.

Table 3.3 Recommend analysis methods and condition by ISO [50]

Type of mooring	Limit state	Conditions to be analysed	Analysis method
Permanent mooring	ULS	Intact/Redundancy check	Dynamic
		Transient [a]	Quasi-static or Dynamic
	FLS	Intact	Dynamic
	SLS	No guidance given	No guidance given
Mobile mooring	ULS	Intact/Redundancy check	Quasi-static or Dynamic
		Transient [a b]	Quasi-static or Dynamic
	FLS	Not required	Not applicable
	SLS	No guidance given	No guidance given

[a] Applicable only if another installation is in proximity to the mooring.

[b] Applicable for MODUs drilling in deepwater where excessive transient motions can cause stroke-out of the riser slip joint.

Table 3.4 Offshore structural design references

Reference	**Title**
API-RP-2SK	Design and analysis of stationkeeping systems for floating structures
API RP-2SM	Recommended Practice for Design, Manufacturing, and Maintenance of Synthetic Fibre Ropes for Offshore Mooring
DNV-OS-A101	Safety Principles and Arrangement
DNV-OS-B101	Metallic Materials
DNV-OS-C101	Design of Offshore Steel Structures, General (LRFD method)
DNV-OS-C401	Fabrication and Testing of Offshore Structures
DNV-OS-D101	Marine Machinery Systems and Equipment
DNV-OS-E301	Position Mooring
DNV-RP-C204	Design against Accidental Loads
DNV-RP-C203	Fatigue Strength Analysis of Offshore Steel Structures
DNV-RP-C205	Environmental Conditions and Environmental Loads
DNV-OS-C201	Structural Design of Offshore Units (WSD Method)
ISO 19901-7	Stationkeeping systems for floating offshore structures and mobile offshore units.

For those not covered in the international ISO standard, the API Recommended Practice 2SK, *Design and analysis of station keeping systems for floating structures* (API RP-2SK [49]) can be incorporated. This document provides extensive

guidance, including criteria not set in ISO 19901-7 [50]. For synthetic fibre rope moorings, API RP-2SM [75], *Recommended Practice for Design, Manufacturing, and Maintenance of Synthetic Fibre Ropes for Offshore Mooring* have addressed the topic.

DNV-OS-E301 [14], *Position Mooring*, can be applied to all sorts of floating offshore units. This standard, in conjunction with DNV-RP-C205 [76] *Environmental Conditions*, is deemed to be an alternative to ISO 19901-7 [50]. All the codes and standards mentioned above are commonly seen and used in offshore mooring design and analysis. These standards are generally reviewed and revised constantly, which means that the references given here may not represent the latest version of that particular standard. Some documents relevant to mooring design and analysis have been summarised in Table 3.4.

3.11 Example of a practical engineering mooring design

This section illustrates a practical engineering mooring design and the related issues, at the conceptual design stage. It is a practical solution to mooring design provided by AMOG. The purpose of giving an example of a mooring design is to explore the design variables and points of interest in an offshore mooring system, as noted by offshore engineers. From this particular example, concerns in terms of designing a mooring system are revealed.

3.11.1 Overview of a turret mooring

A relatively large vessel with a full load displacement of 170,000 tonnes, is to be moored at a shallower water location where the LAT is 65 m. The mooring solution to this project is provided as follows.

The stationkeeping system of the FPSO is designed as a turret mooring to allow weathervane with a symmetrical 3×3 mooring legs configuration, as shown in Figure 3.35. Each leg has the same material and geometrical properties, and every anchor leg comprises an upper chain segment, a middle chain segment, and a ground chain segment, as presented in Table 3.6. Mooring leg number presented in Figure 3.35. Details of the reference system are given below:

Table 3.5 Mooring leg composition

	Units	Top segment	Middle segment	Ground segment
Material	-	Chain	Chain	Chain
Grade	-	R3	R3	R3
Size	mm	116	152	108
Length	m	125	100	360
Mass in Air [1]	kg/m	313	462	233
Notes: 1. Chain assumed to be studless				

- Water depth, LAT = 65 m
- Anchor/mooring pattern, a turret mooring system
- Anchor radius 550 m
- Total offset, 25.5 m in a positive *x* direction
- Number of mooring lines, 9 identical studless chains consisting of three segments of chain in a 3×3 configuration

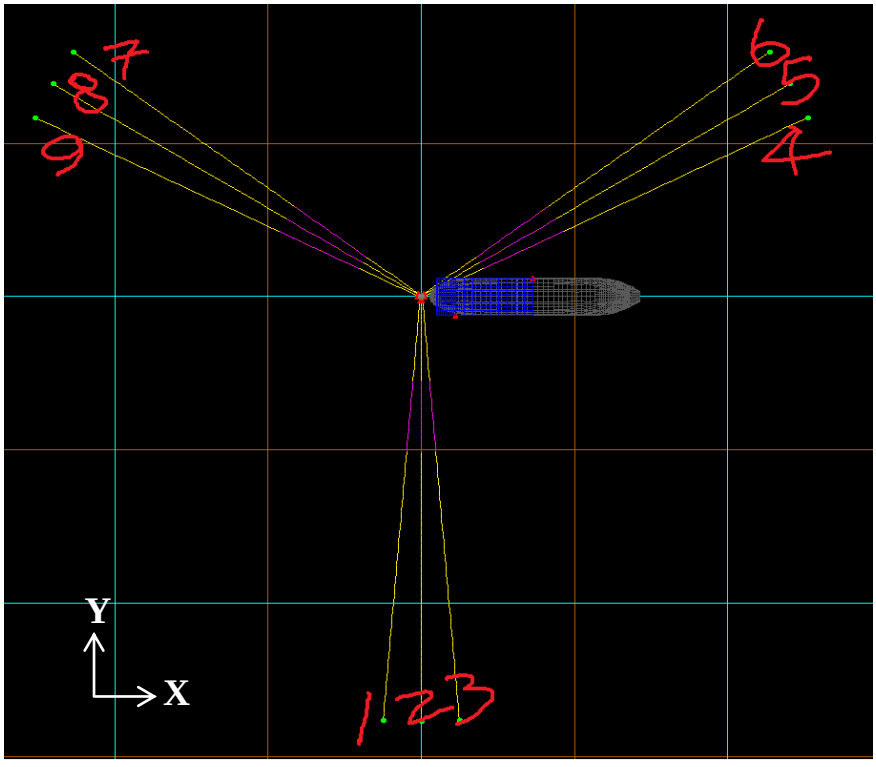

Figure 3.35 Top view of the stationkeeping system

The material and length of each anchor leg are presented in the Table 3.5 [77]. A summary of the mooring line anchor and fairlead positions are presented in Table

3.6. The headings of the anchor line, in degrees, are in relation to due north, which is pointing to the negative direction of the *x* axis. The positive angle is counter clockwise. The corresponding values of the fairlead position *x* and *y* of each mooring leg are presented in global coordinates. The *z* coordinates are measured from the baseline of the FPSO, which is 31.5m above the keel to the base of the chain table (The draft of the vessel is 16 m). The full design data can be found in Marcollo [77], and the mooring system has been modelled in OrcaFlex for simulation.

Table 3.6 Mooring line anchor and fairlead positions

Anchor Line	**Anchor Line Heading (deg)**	**Fairlead Position (global)**			**Anchor Radius (m)**
		X (m)	**Y (m)**	**Z (m)**	
1	85	-1.72	-4.82	31.5	550
2	90	0	-4.87	31.5	550
3	95	1.72	-4.82	31.5	550
4	205	5.04	0.92	31.5	550
5	210	4.22	2.44	31.5	550
6	215	3.32	3.9	31.5	550
7	325	-3.32	3.9	31.5	550
8	330	-4.22	2.44	31.5	550
9	335	-5.04	0.92	31.5	550

3.11.2 Concerns and variables of the Mooring design

This is a mooring case with a large FPSO moored in a relatively shallow depth of water. There are a few reasons for using this special one. First of all, in offshore engineering, the availability of a vessel is the first design priority, because few projects can have exclusive customised vessels. Instead, many of them depend on the availability of vessels from investors stock. Second, the challenge of mooring a large vessel in shallow water is higher than mooring a smaller vessel in the same depth of water, because to keep the same number of offsets, large vessels require more sophisticated mooring systems than smaller ones, hence this case has a high level of difficulty which will expose the challenges inherent in designing a turret mooring system. In Australia, more small and marginal reservoirs are being exploited in relatively shallow waters, and meanwhile the increased application of FPSOs requires the re-use of FPSOs and mooring systems re-designed to fit ‘used vessels’ to the new locations.

The solution to the mooring design determines its composition, such as material, grade of steel, length of mooring line, and the number of segments in the mooring line. There are a few aspects that should be considered while designing this mooring system. First and foremost is the cost of the design, closely followed by the mooring restoring force that ensures the system works, the total excursion of the vessel to keep the risers under control, the capacity of the payload and the anchor, et cetera, are important factors that should also be taken into account. To design a mooring system in section 3.11.1, the variables and constraints can be various. To sum up, all the points of interests in a mooring system include, but are not limited to:

- Mooring patterns
- Mooring material
- Mooring line segmentations
- Mooring line lengths
- Restoring force
- Excursion of the vessel
- Vessel mean position
- Mooring line tensions
- Payload of the system
- Mooring anchor tensions

It is obvious that designing a mooring system involves a few design variables. There are also a number of concerns in a practical system. Of all of the points of interests, some of them are independent, such as the mooring material, and can be fully decided by the designers, while some of them are associated with other variables. All of the independent variables are normally chosen as design variables in optimisation, while the dependent coefficients are normally monitored as constraints.

3.12 Characteristic of the mooring design problem

To determine the most appropriate method of improving the design process, the characteristics of mooring problem are examined. The characteristics of mooring

design problems have been investigated in association with the knowledge review of mooring infrastructure in section 3.3, and the design process given in section 3.9 in a simple turret mooring example shown in Figure 3.36.

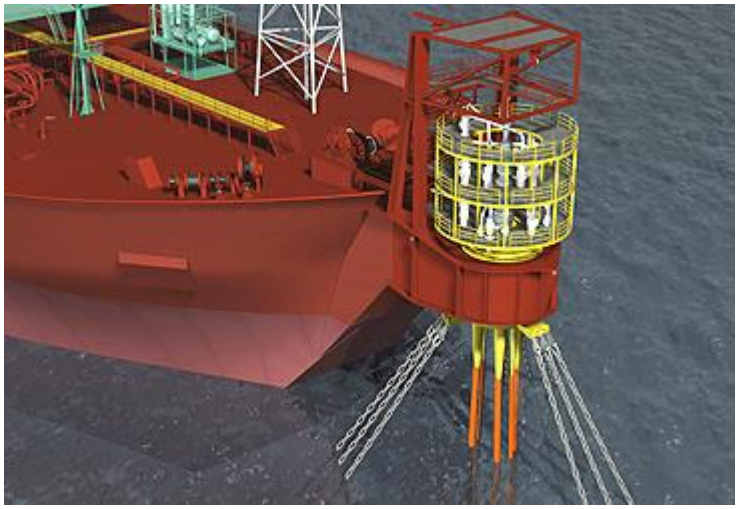

Figure 3.36 An external turret mooring system [78]

A typical mooring system has several parameters that should be considered in the design. These are the strength of the mooring line, the type and positions of the anchors, and environmental conditions and platform requirements. Mooring lines can be made up of various segments with different materials and properties. There are two materials which can be selected: steel chain and fibre rope. If a static equilibrium position is being considered, the variable size of this type of problem would reach a total of 10 or more.

Ensuring that the constraints are complied with is critical in the design and optimisation of mooring systems. Common constraints such as the maximum and minimum length of the mooring, the maximum allowable tension that a mooring line can take before it breaks, and the payload required by the owner are typical examples. Simulated load cases with operational and survival conditions must also be taken into account. Optimising a mooring system means that each hypothesis must be simulated in different load cases under at least two conditions, using time domain or frequency domain analysis. This would mean that hundreds and thousands of evaluations would be needed to fulfil the task.

In general, mooring optimisations have the following characteristics:

- Variable classification. The selection of design variables are project based, so apart from the typical parameters already introduced, other parameters such as mooring patterns and platform headings may need to be considered in different problems. Variables like the length of mooring lines length are continuous, but others such as the type of material and diameters of the chain are discrete. Continuous variables give real valued function optimisation problems, but when combined with discrete design variables, the mooring optimisation problem can be described as a mixed integer problem.
- Constraints. It is confirmed without argument that the mooring optimisation problem is a constrained problem. Take the one shown in Figure 3.35 as an example; the length of the mooring line must greater than the horizontal distance between the vessel and the anchor, the maximum tension cannot exceed the strength of the material strength defined in standards, and the system should be strong enough to keep the vessel in position with certain environmental loads. It is expected that finding a feasible solution is difficult with such a highly constrained problem, and on occasions, due to the amount and limit of constraints, finding a totally feasible solution becomes challenging. Understanding the difference between hard constraints and lenient constraints is important because, for example, the maximum tension that the mooring line has in Figure 3.35 is categorised in the hard constraints group so any violation of this type of constraint is prohibited. However, constraints such as the payload of the vessel, if any, can be grouped into lenient because a small violation of the lenient group can be balanced by sacrificing other aspects of the system, after negotiation.
- Simulation noise. The objective function has a high possibility of being subjected to noise, of which there are two sources. The first stems from evaluating the objective function and simulation, and random and unavoidable noises in the objective function within changes to the design variables. This type of source that contributed from the natural optimisation algorithms is inevitable. The other source is due to the penalty function constructed and aligned with the objective function. The searching process

for the optimisation may trigger violations of different constraints to some degree at different stages, but this numerical noise can be minimised with the well tuned penalty function.

3.13 Chapter summary

This chapter presented the background of offshore mooring that explores essential areas which are relevant to the research of the topic. It presented a review of components, technologies, terminology, and the software package OrcaFlex. OrcaFlex is leading offshore simulation package in both the statics and dynamics of offshore systems. It can handle vessels, moorings, and risers in a complex system, in a time domain manner. Therefore, the results provided by OrcaFlex are a fully coupled solution when the correct options are selected. However, giving coupled solutions can be very time consuming in some circumstances, so any further interpretation of design simulation still requires close involvement with experienced offshore engineers. Both the merits and limitations have been discussed and the terminations of mooring systems, such as the types of anchors, have also been illustrated. Material and its strengths, with regards to different sorts of moorings have also been demonstrated with examples and graphs. This review has been conducted with an Australia bias, and outlines offshore engineering in Australia, and its importance around the world.

As a prerequisite for designing, analysing, and optimising an offshore mooring system, this chapter has given a fundamental overview of the mechanics, methodologies, and engineering practice required for an offshore mooring system. The mechanics of mooring analysis shows that a mooring cable has a catenary property. The design of offshore mooring structures is a difficult task which involves a number of options such as variable definition, analysis, and evaluations, and therefore optimising the mooring design has been considered necessary.

One outcome from analysing the mooring lines leads to improvements in the flexibility iteration method. The slack taut algorithm extended the suitability of the flexibility iteration method in multi-component cable configuration. An example has

been given and the results were compared with OrcaFlex. A practical engineering mooring project has been given with an analysis of the variables, concerns, and points of interests in the design. This chapter finished by giving the characteristics of mooring design problems. It involved a number of complexities and uncertainties which carried the mooring design a step further. Therefore, mooring optimisation is considered to be necessary. The next chapter introduces optimisation techniques that can potentially be applied to the mooring optimisation.

4 GENETIC ALGORITHM & PARTICLE SWARM OPTIMISATION

4.1 Introduction

Particle swarm optimisation (PSO) is a heuristic algorithm inspired by the choreography of a swarm of bees. It was first introduced by Kennedy and Eberhart [79]. PSO is based on the premise that individuals share their information within the swarm for the purpose of seeking advantages (goal). It has been found to be successful in a variety of ways in engineering applications like composite beam design [80], the design of logic circuits [81], control design [82, 83] and concrete beam design [84]. When compared with other optimisation algorithms, Hassan [85] and Shi [86], claimed that PSO requires a fewer number of evaluations, and gives reliable results.

Genetic Algorithm (GA) is an optimisation method based on the famous rule of 'Survival of the fittest' from Charles Darwin, an English naturalist. It simulates the random selection process of nature. Information that fully covers GA can be found in the Refs [87, 88] and discussed below.

In this chapter, two of evolutionary optimisation techniques GA and PSO are reviewed in terms of concept, mathematical fundamental processes and the employment for benchmarking in the next chapter. Both algorithms are stochastic in nature and do not require derivative calculations. It serves as footstone in applying evolutionary optimisation into offshore mooring design.

4.2 Genetic algorithms (GAs)

The main reason for categorising GA as evolutionary optimisation is its ability to escape local optimum, and hence avoiding the problem inherent in a number of classical optimisation methods. GA deals with a set of potential solutions called population that resembles a string of chromosomes, thus the representation is called genotype. The unit of a genotype is called a gene, which represents the values of each design variable to a problem. There are various notations of representative chromosomes, like a length of string, floating points, and q-ary coding [89].

GA is comprised of three stages, including an evaluation of the fitness function, a selection of the fittest individuals, and reproduction of the offspring [90]. In the first phase, the fitness evaluation, the objective function is calculated for every population in a generation. In the selection phase, the possibility associated with individual to be selected as reproductive parents is related to their fitness values. Generally, the higher the value of fitness assigned (it means they are better at their objective performances), the greater is its possibility of being selected to reproduce. The reproductive phase has two operators in function, the first is the crossover and the second is the mutation. The two operators create the next generation according to the formula

$$\begin{aligned} x &= \gamma x_{parent1} + (1-\gamma) x_{parent2} \\ x &= \alpha\beta\gamma x_{parent1} + \alpha\beta(1-\gamma) x_{parent2} \end{aligned} \tag{4.1}$$

where x represents the design variables, γ is a random number that determines how many genes are needed from that parent, α is the possible mutation, and β is a random number between 0 and the length of the chromosome needed to decide which gene is to be mutated.

The processes of running GA's are as follows:

1. Randomly initialise the first population (if some potential solutions are known beforehand they could be used in the population).
2. All the populations go to the selection operator with their corresponding fitness output, and are meanwhile assigned the possibility of reproduction.
3. Crossover and mutation operators come in place to reproduce the next generations.
4. Repeat steps (2) and (3) till the convergence criteria is met, for example, the fitness value is not changing in a certain number of iterations.

A flow chart of running GA's has been summarised below by Fenwick [19]. When programming GA's in software packages, the pseudo-code can be expressed as follows:

WHILE stopping criteria reached

FOR each population in a generation, calculate their fitness values based on an objective function

SELECT best parents to crossover

SET mutate rate to allow for random change of chromosomes

END

END

END

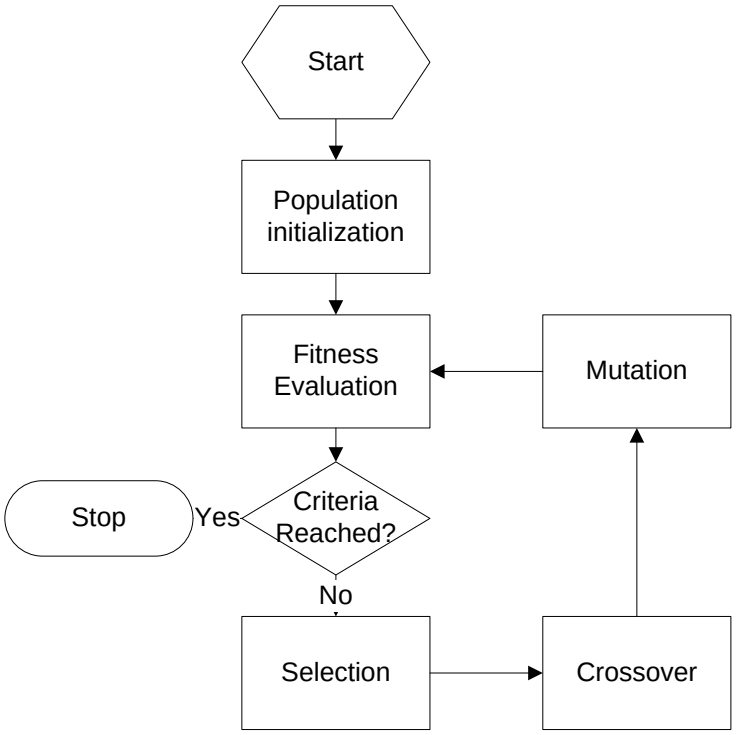

Figure 4.1 GA flowchart

4.2.1 Genetic algorithm operators – roulette wheel selection

As shown in Figure 4.1, there are three operators in GA, called selection, crossover, and mutation. The selection operator is the one to decide which 'parents' are allowed to reproduce 'offspring' on the basis of their fitness. Of all the common selection techniques, such as stochastic universal sampling, tournament selection, proportionate, and ranking, etc., the roulette wheel selection (RWS) is the one most commonly used. A comprehensive detail of the different selection algorithms can be found in Bäck [91].

The probability of an individual being evaluated on the basis of fitness is represented by the proportion of a wheel as shown in Figure 4.2. The occupation of each sector of a roulette wheel represents the probability of each individual's allocation according to their fitness values. The total area of sectors is represented by the cumulative probabilities of the individuals proceeding to it, and therefore the total probability of the whole wheel is equal to one.

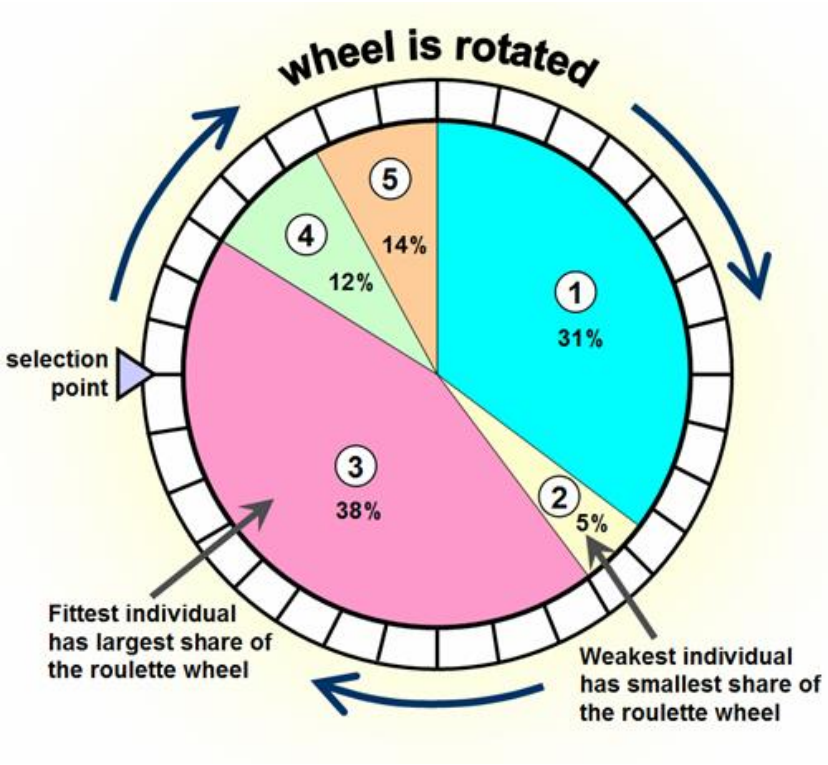

Figure 4.2 Roulette wheel selection diagram [43]

A random number between 0 and 1 that mimics the process of a wheel spinning is then generated and the individual that corresponds to the number generated is selected for reproduction. This process is repeated a number of times on the basis of the number of individuals required. However, Fenwick [19] mentioned that it is possible that selected individuals are disproportionate to their probability due to the nature of random sampling. Finally, the selected individuals are placed in a 'gene pool' awaiting crossover.

4.2.2 Genetic algorithm operators – tournament

Tournament selection is a two stage algorithm that involves selections and comparison. First of all, a number of individuals (normally two) are selected from the populations to be tournament candidates. This process can however, be fully customised due to selection requirements. Selecting the tournament candidates is a random process where the number of winners can also be adjusted and each candidate can be compared with the best one or two (depending on the number of winners), being selected for reproduction.

Since the tournament algorithm requires no extra computational cost in ranking and scaling, the advantage of tournament selection in parallel computation is obvious [91]. In a parallel processing environment, independent variables are required and preferred to fulfil tasks without interference so the grouping nature of tournament selection fits the requirement with confidence and convenience. As a result the tournament selection algorithm can be used in situations where full evaluation of an individual is not necessary.

4.2.3 Genetic algorithm operators – crossover

A crossover operator is an algorithm used to swap genetic information between parents for the purpose of reproducing offspring. Even though the possibility of having more than two parents to generate offspring exists, only two parents are generally needed to produce offspring.

Essentially, crossover is the process of replacing some of the alleles of an individual by genes from the other individual using techniques such as one-point crossover, two-point crossover, and uniform crossover, etc [89]. Associated with all the crossover techniques, bit string representation is among the most traditional method. A schematic view of a bit string representation with one and two-point crossovers are shown in Figure 4.3. The formulation of children is given in Equation (4.1).

The number of crossover points can be set consistently over the whole process, although it can also be various. The location of the crossover point is randomly generated, and all of these parameters are adjusted on a case by case basis.

4.2.4 Genetic algorithm operators – mutation

The existence of a mutation operator is a crucial to increase the variability and feasibility. The concept of mutation is to change the gene of an individual at random in a pre-set probability. For instance, in a binary representation the mutation operator changes a bit from 0 to 1 or from 1 to 0 if the algorithm is being trigged. Trigging by the mutation operator is heavily dependent on the mutation rate set for this algorithm. According to Whitely [92], a percentage of one is a typical mutation rate that would enable the regions of a search space to be thoroughly explored. In a q-ray mutation the gene is changed to any of the possible q-ray values.

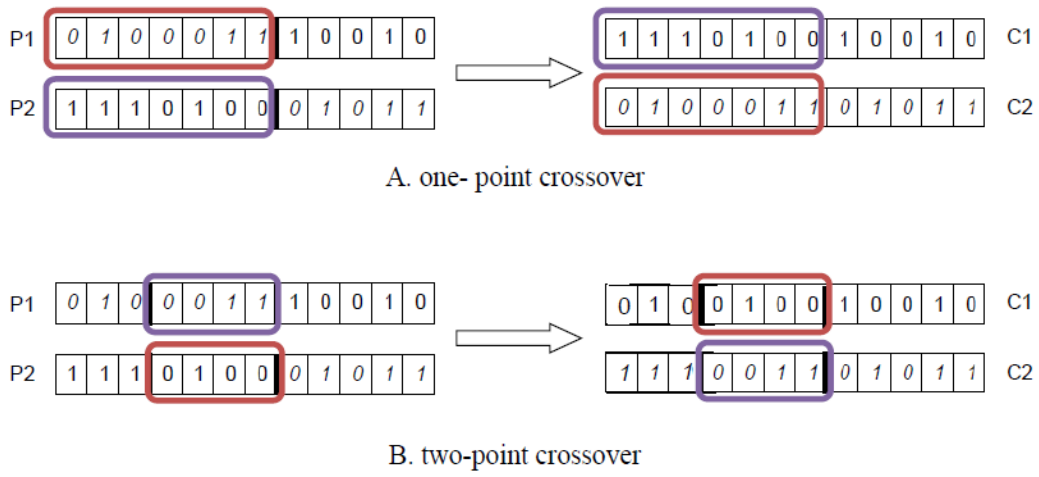

Figure 4.3 Crossover techniques

The purpose of applying a mutation operator in GAs is to widen the exploratory search space so that the algorithm can escape from the local optimum. A genetic algorithm without a mutation operator may cause problems such as premature convergence and limited diversity. However, as the mutation rate increases, a whole population evolves in a random pattern without improving the quality, with the result that at a certain mutation rate, a better solution space with optimum answers are more likely to be located by GAs.

4.3 Description and mathematical characteristics of PSO

Kennedy and Eberhart [79] first introduced an iterative based optimisation technique which they developed by simulating the social behaviour of a flock of birds or a

school of fish, and called particle swarm optimisation (PSO). It is similar to GA in that the fitness of the objective function is evaluated on an individual basis, although termed 'particle' in PSO. After each iteration the movement of the particle is influenced by its best position from its local position, while being guided towards the best position in the swarm.

The advantage of this algorithm is that it allows for the utilisation of 'prior knowledge' in the search process while information about the local particle and swarm are shared in order to direct the trajectory of the swarming. The algorithm terminates when the swarm approaches the best known position regardless of the behaviour of the swarm, or the whole swarm converges to a position in the solution space.

Like other evolutionary algorithms introduced in Chapter 2, PSO is based on the population of random solutions, called particles, which are in essence, candidates from the solution space. Each particle has a velocity which allows them to move throughout the domains. The velocity vector is updated through the historical behaviour of the particles. To be specific about the velocities, they are determined by the history of the best position. There are two types of positions in PSO, the best personal position which is referred to as the best local position, and the best swarm position which is the best global position. The best personal position is the best solution that each individual particle has ever had in its moving trajectory. The best swarm position (sometimes called the best global position) is the best solution selected from all the particles.

Mathematically, the position x of a particle i, at time t is updated as

$$x_t^i = x_{t-1}^i + v_t^i \Delta t \tag{4.2}$$

where v_t^i is the velocity vector at time t, and Δt is the time step during two iterations. Figure 4.4 shows how the particle positions are updated.

The 'time' incremental step Δt is taken here as a unit for the convenience of calculation. Particles have set values for each of the input variables, which have been defined as the dimensions of the solution space. The velocity component of a particle

has been separated into vectors that have multi-dimensions. Each of the dimensions in the velocity represents the changing of a variable. The velocity vector of each particle is calculated as

$$v_t^i = wv_{t-1}^i + \frac{c_1 r_1 (p_{t-1}^i - x_{t-1}^i)}{\Delta t} + \frac{c_2 r_2 (p_{t-1}^g - x_{t-1}^i)}{\Delta t} \tag{4.3}$$

where both r_1 and r_2 are generated uniformly between 0 and 1 for the purpose of providing randomness; p_{t-1}^i corresponds to the best position for particle *i* in its time history; p_{t-1}^g represents the best position for all the particles at time *t-1*; c_1 and c_2 are parameters showing the confidence the particle has in itself and among the swarm, respectively; *w* is the coefficient of inertia.

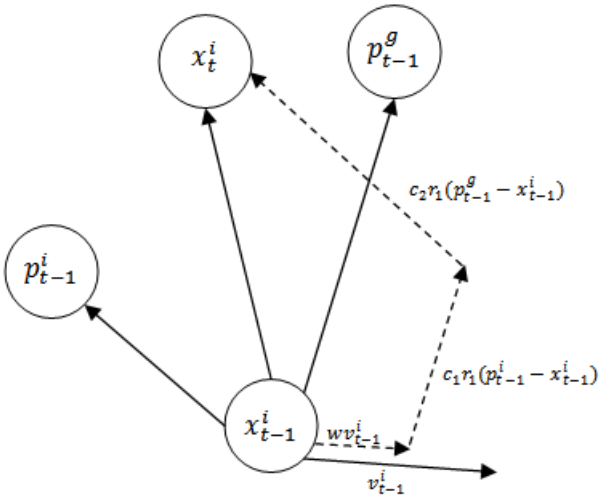

Figure 4.4 A schematic view of a particle position updates

Substitute Equation (4.3) into (4.2) updates to get the following position expressed as:

$$x_t^i = x_{t-1}^i + \left(wv_{t-1}^i + c_1 r_1 \frac{(p_{t-1}^i - x_{t-1}^i)}{\Delta t} + c_2 r_2 \frac{(p_{t-1}^g - x_{t-1}^i)}{\Delta t} \right) \Delta t \tag{4.4}$$

Re-arrange Equation (4.4) in terms of position *x* and velocity *v* and then combine them in the form of a matrix, which leads to Equation (4.5):

$$\begin{bmatrix} x_t^i \\ v_t^i \end{bmatrix} = \begin{bmatrix} 1 - c_1 r_1 - c_2 r_2 & w\Delta t \\ -\frac{(c_1 r_1 + c_2 r_2)}{\Delta t} & w \end{bmatrix} \begin{bmatrix} x_{t-1}^i \\ v_{t-1}^i \end{bmatrix} + \begin{bmatrix} c_1 r_1 & c_2 r_2 \\ \frac{c_1 r_1}{\Delta t} & \frac{c_2 r_2}{\Delta t} \end{bmatrix} \begin{bmatrix} p_{t-1}^i \\ p_{t-1}^g \end{bmatrix} \tag{4.5}$$

According to Perez [93], the process of PSO can be summarised into the following steps:

1. Initialise a set of particles distributed randomly through the solution space, where the initial velocity can be either zeros or random values.
2. Evaluate the swarm of particles on the basis of the objective function of a problem and store the information about their position.
3. Update the position of each particle and corresponding velocity through their previous information.
4. Repeat steps (2) and (3) until the stop criteria has been reached, for example, the repeating number of iterations.

When programming PSO in software packages, the pseudo-code for it is as follows:

WHILE stopping criteria reached

FOR each particle in a swarm, calculate their fitness values based on the objective function

FIND the velocities for each particle

IF the fitness value is better than the best fitness memorised in the history

SET 1. the best fitness value to the current better value and stored

2. new particle positions

END

END

END

4.3.1 PSO parameters

There are common parameters such as the size of the candidate and stopping criteria that share the same characteristics with all the evolutionary optimisation approaches. In PSO, the size of the candidate is the number of particles in the swarm. Larger numbers of particles explore a greater response surface at each iteration and thus have a higher possibility of finding the global optimum. However, a swarm of excessive size can result in a parallel random search and an increase in the computational time. A swarm size of 10 to 30 has been found from empirical studies by Brits [94] and Van den Bergh et al. [95] to be appropriate. Nevertheless, the best

number of swarm size depends on the optimisation problem as this parameter was found more or less on a trial and error basis.

The acceleration coefficients c_1 and c_2 control the random search of the cognitive and social components of velocity. The exploratory nature of particles is determined by their relative values. A large cognitive acceleration coefficient makes the particles wander excessively, while a large social acceleration coefficient tends to trap the optimisation in local minima. Referring to the PSO stability analysis in Perez and Behdinan [93] from the discrete dynamic system in Equation (4.5), w, c_1 and c_2 are claimed to be stable as long as Equation (4.6) is met.

$$\begin{aligned} 0 < c_1 + c_2 < 4 \\ \frac{c_1 + c_2}{2} - 1 < w < 1 \end{aligned} \tag{4.6}$$

There are a few discussions about those coefficients. Instead of using the weight of static inertia, Eberhart and Shi [96] claimed a dynamic improvement can be made to the weight of inertia by using a constriction factor. Ratnaweera [97] proposed a method that gave a higher initial value of c_1 but then reduced it at each iteration, while c_2 had a low initial value and increased at each iteration. They are given by:

$$\begin{aligned} c_1(t) &= \frac{(c_{1,\min} - c_{1,\max})t}{n_t} + c_{1,\max} \\ c_2(t) &= \frac{(c_{2,\max} - c_{2,\min})t}{n_t} + c_{2,\min} \end{aligned} \tag{4.7}$$

where $c_{1,max} = c_{2,max} = 2.5$ and $c_{1,min} = c_{2,min} = 0.5$, t is the time step and n_t is the total number of time steps. This method initially allows particles to explore the search space widely, and then converge to a good optimum towards the end of the process. The condition for controlling stopping is the maximum number of iterations.

Another important parameter is the stopping criteria which governs the maturity of the PSO. Often when running the PSO, there is a set of iterations that show no improvement, and then one particle attains a better position. Therefore, early termination will give a sub-optimal solution and late termination will take up extra computational time. The stopping criteria includes but is not limited to:

- The iteration number specified by users. This number cannot be too small to get sub-optimal results and it cannot be too large to increase the computational costs either.
- The value of an objective function is not improving over a certain number of iterations. This indicates very small changes in particle positions over a number of iterations, or no distinct improvement over a certain number of iterations.
- A certain proportion of particles are clustered [95]. This means that the swarm is clustered around a point and do not change positions over a certain number of iterations.
- The objective function has an approximately zero slope. The slope defined by Van den Bergh [95] is expressed as

$$f' = \frac{f(y(t)) - f(y(t-1))}{f(y(t))} \tag{4.8}$$

4.3.2 Handling particles outside the viable solution space

The positions of particles are updated via their velocities, but large velocities have the potential to drive particles outside the solution space. The method for dealing with this violation of particles introduced by Venter and Sobieszczanski-Sobieski [98] restricts the velocity vector of a violated particle while it tends to travel to infeasible space. The tactic used in ref [98] was to minimise the effect of momentum on the particle by making the coefficient of inertia zero, so the velocity Equation (4.3) changed to

$$v_t^i = c_1 r_1 \frac{(p_{t-1}^i - x_{t-1}^i)}{\Delta t} + c_2 r_2 \frac{(p_{t-1}^g - x_{t-1}^i)}{\Delta t} \tag{4.9}$$

In Equation (4.9), the velocity of the particle is only affected by the best global position and its best position from the history record. The new velocity vector will point the particle back to the feasible region of design. Therefore, the velocity of particle *i* at time *t* is only influenced by the best global position for the particle in the swarm and the best position found for that particle so far. If, fortunately, both positions are feasible, the new velocity vector will re-direct the particle to a feasible region of the design space. But if either position is in the feasible zone, or neither of

them locates in the feasible zone, the new velocity vector will re-direct the particle to a region with a smaller violation.

Kathiravan and Ganguli [80] used another strategy in the optimum design of a composite box beam structure in strength constraints. Their strategy controls the particles in the solution space by limiting their maximum speed. Their method gave a smooth change to the position of the particles.

The methods introduced above constrain the particles and tends to move them to infeasible regions, but if the particles are already outside the viable solution space, those techniques cannot work anymore. At the position where outside the viable solution space, the death penalty (introduced in section 4.5.2) can be triggered and the particle can be eliminated. It is possible that the deathly penalised particle information still wants to be maintained for further processing, such as repairing. However, the objective function cannot be evaluated due to variable domains being exceeded, so a more sophisticated scheme is required to handle that situation.

In that circumstance, the global optimum has a high possibility of being around the boundary of the search space so that the particles need to be pulled back into the viable solution space. This can be done by relocating the particle to its nearest viable solution boundary. When the position of any particle has been detected as outside the viable solution space, an interfering algorithm gets to modify the position information so that the particle can be relocated to its nearest boundary. Nevertheless, the pulling force cannot be too large otherwise it will ruin the search completely.

This scheme has been proposed in Figure 4.5 because it gives a more thorough and sophisticated search near the boundary for optimum. In the diagram, the normal arrow shows the velocity vector due to the combined PSO velocity, and the bold arrow demonstrates the pulling force that drags particles from the infeasible design space to the viable solution space.

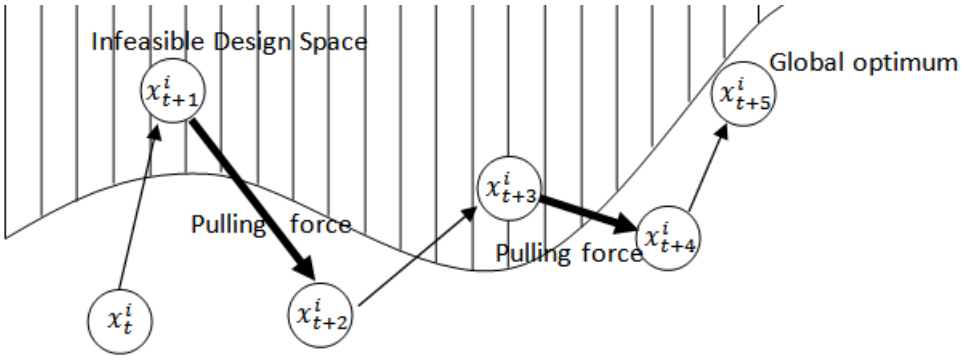

Figure 4.5 Schematic view of pulling violated particles back to feasible design space

It is quite common for the final optimum to lie near the edge of the constraint boundaries of the space of feasible solutions [90]. Pulling particles which are outside of the solution space back to the vicinity of their boundary has a few advantages:

- The search path of the algorithm is not changed via enforcing particles back to the search space
- It gives a thorough search near the boundaries where optimisation solution has a high possibility of occurring
- It does not re-direct particles to different directions.

4.4 PSO and related works

Since the establishment of the PSO by Kennedy and Eberhart [79] in 1995, the algorithm itself has been improved in terms of capability, convergence speed, and generality. When reviewing the progress of PSO, the author found over 1000 articles in IEEE. This number shows the increasing interest of researchers and engineers in this optimisation algorithm. Wang [99] stated that all of those improvements promote the use of PSO in many engineering application.

In recent years several of the very latest PSOs have been proposed to advance its features for global optimisation. Comprehensive learning PSO (CLPSO) by Liang [100] is designed for multi-modal problems. When this method is used for a uni-modal problem its converging speed is much lower. Cooperative based PSO (CPSO) developed by van den Bergh and Engelbrecht [101] improved the performance of PSO by cooperatively using multiple swarms in components from the solution

vector. However, the authors still claimed that it was unclear and they had not proven that the approach would be better for all problems. Hsieh and Sun [102] used a population manager to improve the efficiency of PSO, and called their method the efficient population utilisation strategy based PSO (EPUS-PSO). The main idea with this approach is the use of variable particles in swarms. This approach performed well in many bench marking problems but the authors did not try it out on any practical engineering problems. As with the EPUS-PSO, the fully informed PSO (FIPS-PSO) introduced by Mendes et al. [103] had not demonstrated the ability to solve real engineering problems, even though it extended the working of PSO in ten dimensions, which could not be achieved very easily by EPUS-PSO. Table 4.1 summarises a small portion of the literature on the application of PSO.

Although many works related to a huge variety of PSO have been done, there is no universally acknowledged method of dealing with diverse problems with different characteristics. In the view of applying PSO, many of its improvements are at the stage of bench marking mathematical functions. PSO has a great application in different areas of engineering applications, but only a few extend the great algorithm to offshore field.

Table 4.1 Summary of PSO literature review

Method	Author	Year	Application
PSO	G. Venter et al [104]	2005	Transport aircraft wing optimisation
	Kathiravan, R. et al [80]	2007	Composite beam design
	McCluskey [84]	2008	Reinforced concrete beam design
	Kameyama [105]	2009	Benchmark functions
	Di Giampaolo et al[106]	2010	RFID-Network planning
	de Pina [107]	2011	Oil production risers
Cooperative based PSO (CPSO)	van den Bergh, F. et al [101]	2004	Benchmark optimisation problems (both uni- and multi-modal functions)

Fully informed PSO (FIPS-PSO)	Mendes, et al [103]	2004	Benchmark functions
Improved PSO	Li, Gui and Yang [108]	2005	Power supply
Comprehensive learning PSO (CLPSO)	Liang [100]	2006	Multimodal test functions
Efficient population utilisation strategy based PSO (EPUS-PSO)	Hsieh, et al [102]	2009	Uni- and multi- modal function such as quadric, griewanks, rastrigin, ackley, and weierstrass,

4.5 Multi-Objective PSO

According to Kennedy [109], the application of multi-objective PSO (MOPSO) in engineering is particularly suitable due to the speed of its convergence. The use of the evolutionary multi-objective optimisation (EMO) has grown significantly in the last few years, as claimed in ref [110]. Applying this multi-objective technique into an evolutionary search method may or may not be suitable for different search algorithms because they do have their individual strengths and weaknesses.

4.5.1 Approaches to multiple objectives

Chen and Lu [111] claimed that real problems often require an optimisation that considers the nature of several objectives; for instance, purposes such as cost, performance, and reliability are common objectives. Boulougouris [112] confirmed Chen in the multi-objective optimisation when designing an LNG terminal. By setting an objective utility function without enough knowledge of the design space or feasible solution zone can lead to engineers preferring not to use optimisation, and therefore a single objective is sometimes not enough.

Instead of a single scalar, for which a minimum value can easily be drawn from comparison, a multi-objective algorithm produces a set of good trade off solutions from which decision makers can choose. There are two main approaches to multi-

objective optimisation: namely, aggregation and the Pareto based approach, respectively.

The aggregation approach merges the objective function score vector as a utility function by multiplying a weight factor which reflects the relative importance of that objective. This approach allows for the existence of multiple criteria, but its disadvantage is the requirement for weight factors to be tuned after repeated trial and error in order to achieve the desired optimum. Another limitation becomes prominent when there are conflicting constraints. Even strategies such as the one developed by Le Huédé [113] provided trade offs between criteria by diversifying the search strategies on several Branch & Bound search iterations, without knowing beforehand that the criteria cannot guide searches to satisfying solutions.

The Pareto approach is based on the concept of Pareto dominance, which has been borrowed from the economics. In Pareto dominance, one individual cannot be better off without making another individual worse off. In multi-objective optimisation, the notion of optimality is not at all obvious. There are four basic concepts that need to be pre-defined.

Definition 1 (Pareto dominance). A vector $\vec{X} = (X_1, ..., X_k)$ is said to dominate another vector $\vec{Y} = (Y_1, ..., Y_k)$ (denoted by $\vec{X} \prec \vec{Y}$) if and only if $\vec{X}$ is partially less than $\vec{Y}$, i.e., $\forall i \in \{1, ..., k\}: \; X_i \leq Y_i \wedge (\exists j \in \{1, ..., k\}: \; X_j \leq Y_j)$.

Definition 2 (Pareto optimality). A solution $\vec{x} \in \Omega$ is said to be Pareto optimal with respect to Ω if and only if there is no $\vec{y} \in \Omega$ for which $\vec{Y} = F(\vec{y}) = (f_1(\vec{y})), ..., f_k(\vec{y}))$ dominates $\vec{X} = F(\vec{x}) = (f_1(\vec{x}), ..., f_k(\vec{x}))$.

Definition 3 (Pareto optimal set). The Pareto optimal set P_S is defined as the set of all the Pareto optimal solution, i.e., $P_S = \{\vec{x} \in \Omega \mid \neg \exists \vec{y} \in \Omega: F(\vec{y}) \prec F(\vec{x})\}$.

Definition 4 (Pareto optimal front). The Pareto optimal front P_F is defined as the set of all objective functions values corresponding to the solutions in P_S, i.e., $P_F = \{F(\vec{x}) = (f_1(\vec{x})), ..., f_k(\vec{x})) \mid \vec{x} \in P_S\}$.

Suppose an optimisation job wishes to minimise both objective one and two, both of which are equally important. If there are five possible ways to finish the job (scenarios A, B, C, D, and E), this will result in the following values:

A = (2, 9) (objective 1 and 2)

B = (4, 6)

C = (8, 3)

D = (9, 7)

E = (7, 8)

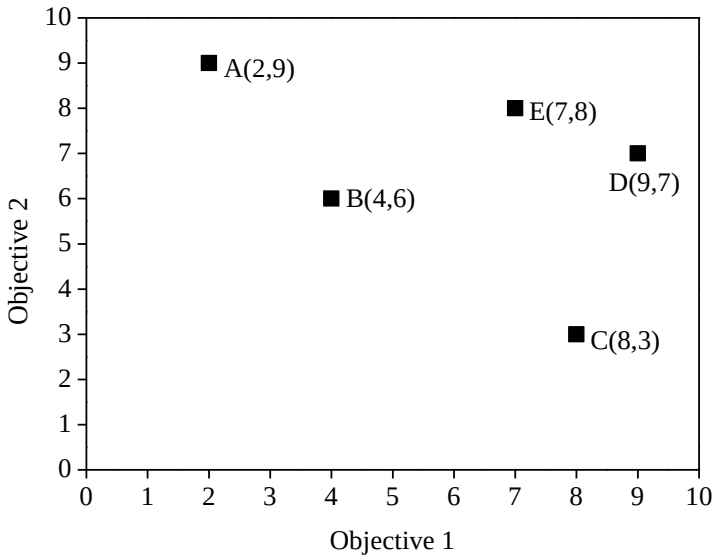

Figure 4.6 Illustration of Pareto dominance

These data points are plotted in Figure 4.6 , a graph of objective one versus objective two. Scanning this graph reveals that best particles are located in the lower left corner. Scenario D is dominated by B and C, because 4 < 9, 8 < 9 and 6 < 7, 3 < 7. Likewise, scenario E is dominated by B. Therefore, scenarios A, B, and C seem to be good possible choices to be selected as global optimum even though none of them is best along both objectives. In optimisation terminology, A, B, and C are non-dominated particles while D and E are dominated particles.

Combine the illustration in Figure 4.6 and definition 4, the set of A, B, and C is the Pareto-optimal front. Thus, in multi-objective problems where a set of solutions are not dominated by any others, the Pareto optimal set should have the prior possibility of sharing their information as optimum.

4.5.2 Pareto approach

Unlike a single objective PSO, where particles can easily be selected by comparing the scalar objective values, the MOPSO requires the assistance of the more complex Pareto dominance. One of the biggest challenges to MOPSO claimed by Knowles [114], is balancing the diverse maintenance ability and optimum global searching efficiency.

MOPSO related works extended PSO to handle multi-objectives on the basis of the Pareto approach. Fieldsend and Singh [115] used a special data structure named 'dominated tree' to store the Pareto optimality set for the search process. Apart from the data structure, this approach used a mutation operator to calculate the velocity, and it was called a 'turbulence' operator. Mostaghim et al [116] improved the convergence and diversity of the MOPSO by adopting a sigma method. This approach resembles the compromise programming proposed by Coello Coello [110].

In recent years the MOPSO techniques [117-120] are more or less related to the employment of an external repository that shares particle information to guide their flights. The external repository does not change the main PSO algorithm and only adds an external information pool (or pools) which stores Pareto front information to guide the search trajectories.

According to Coello Coello [118], the main process of applying the external repository in MOPSO have been summarised below:

1. Initialise the number of particles in a swarm
2. Given the initial speed of each particle, this can normally be set as zero
3. Evaluate each particle in the swarm

4. Memorise the positions of the non-dominated particles in the external repository
5. Generate hyper cubes based on a current Pareto front so far, and determine their coordinates according to their objective values
6. Initialise the historic memory of each particle
7. Compute the speed of each particle via

$$v_t^i = wv_{t-1}^i + c_1 r_1 (pbest^i - x_{t-1}^i) + c_2 r_2 (REP[p] - x_{t-1}^i) \quad (4.10)$$

where the representation of parameters are shown in Equation (4.3); $pbest^i$ is the best position that the particle i has ever had; $REP[p]$ is a global best particle selected from the external repository introduced lately.

8. Compute the new position via Equation (4.4)
9. Update the Pareto front Repository
10. Repeat steps 7 to 9 until the stopping criterion be triggered.

When the current position of each particle is better than its previous position in its history, the $pbest^i$ is updated through Equation (4.11).

$$pbest^i = x^i \quad (4.11)$$

The criterion of a single objective PSO *pbest* selection is the value of its objective function. While in MOPSO, if the position of the current particle is dominated by its memory position, then the memorised position is kept (case 2 in Figure 4.7), but if the current position dominates the memory position, the current position is selected (case 1 in Figure 4.7). Otherwise, (both of the solution sets are Pareto optimal) the best position is selected randomly (case 3 in Figure 4.7).

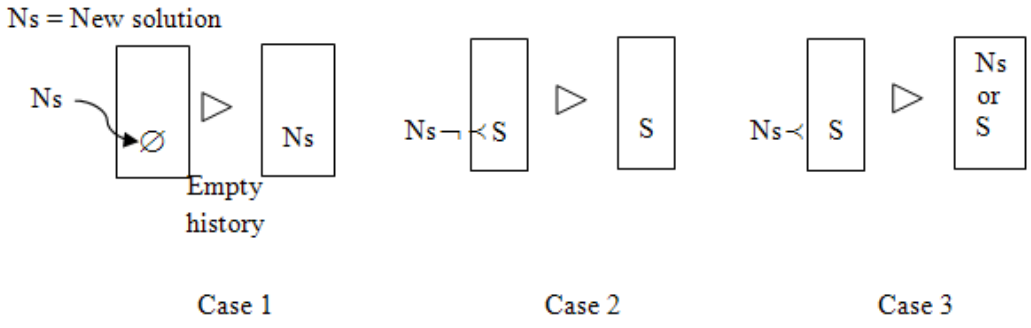

Figure 4.7 Particle history update diagram

The selection criterion for deciding the best particle position historically is based on the Pareto dominance theory. However, selecting the best particle in a swarm from

the repository requires further handling techniques. Coello Coello [118] proposed a hypercube method with a roulette wheel selection. This method gridded the repository through a variation of the adaptive grid proposed by Knowles [114]. The basic idea is to use an external archive to store all the Pareto optimal information with respect to objectives.

The repository building algorithm resembles the process of updating the particle history shown in Figure 4.7. The difference is that the external repository stores all the Pareto optimal solutions from all the particles in a swarm for the whole iteration. Apart from case 1 to case 2, there are two more cases, as shown in Figure 4.8, with regards to all the particles. Case 4 shows how the new solution dominates the existing solution in the archive, and how the existing solution is replaced by a new one. Case 5 invokes the adaptive procedure of the external repository. The third case in Figure 4.8 is slightly different from Figure 4.7. The *pbest* only accepts one particle as the Pareto optimal while the external repository accepts all Pareto Optimal. New solutions can be stored in the repository until maximum capacity has been reached.

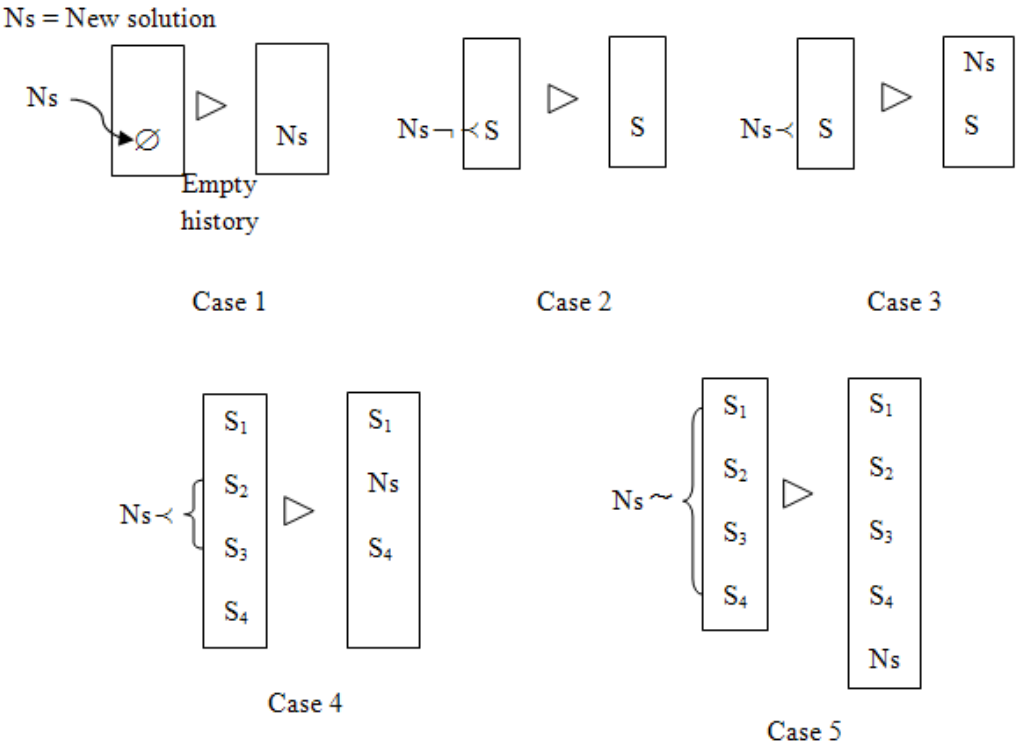

Figure 4.8 Selection algorithm for external repository

The adaptive grid is based on hyper-cubes which may be either two dimensional or multi-dimensional. Each dimension in a hypercube represents an objective function. Each grid in Figure 4.9 and Figure 4.10 equals a hypercube that can be interpreted as a geographical region.

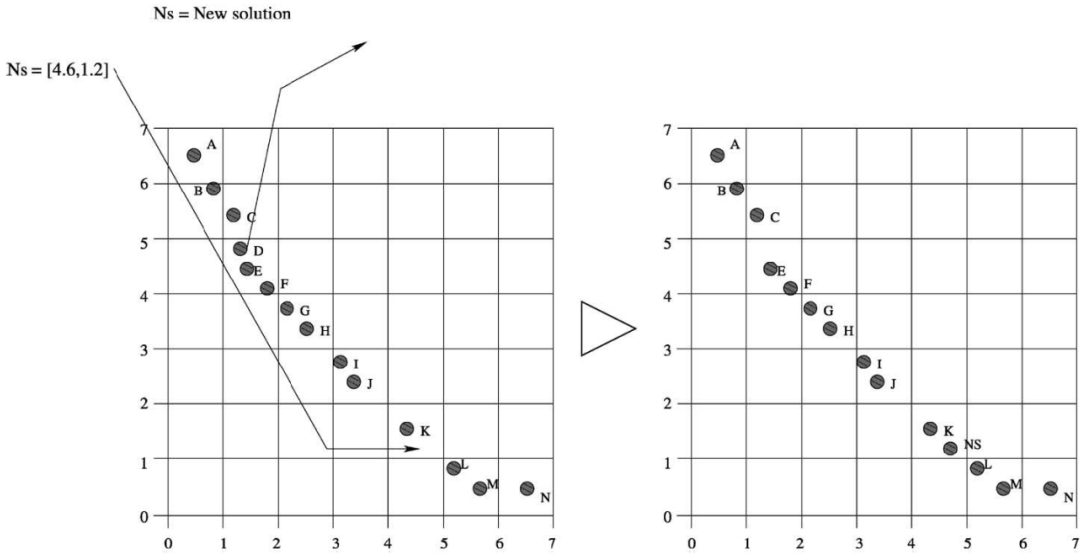

Figure 4.9 Representation of insertion of a new particle in repository [118]

The external repository is divided into a grid, as shown in Figure 4.9. All the new solutions are stored in the repository until it reaches maximum capacity. If the new solution to be stored in the repository is located outside the boundary of the grid (shown in Figure 4.10), the repository will increase its size to re-fit the complete range of the Pareto optimal and the existing grid will also be recalculated.

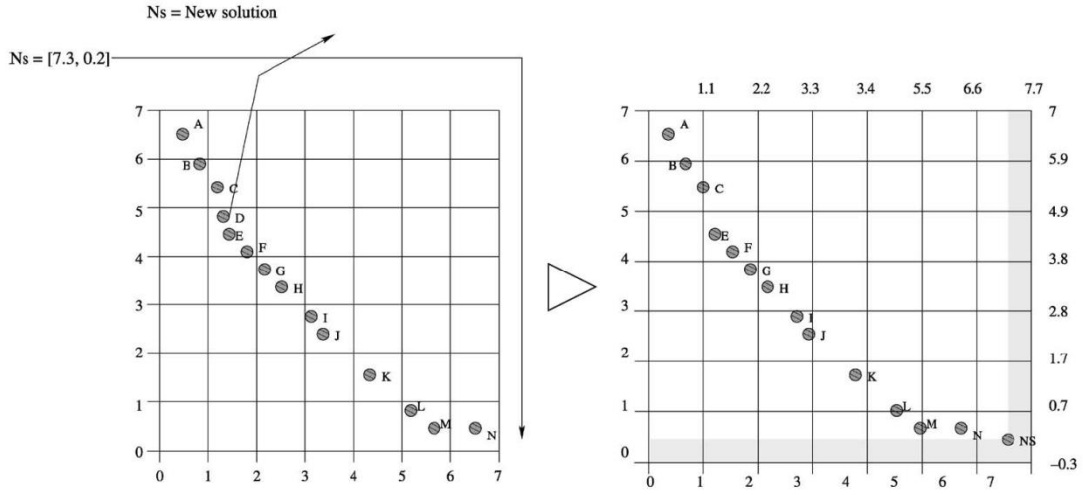

Figure 4.10 Representation of insertion of a new particle in the adaptive grid [118]

The main function of the external archive (or repository) is to maintain a good historical record for the Pareto optimal particles found along their trajectories. Selection of *REP[p]* in Equation (4.10) involves a deep understanding of the geographical representation of the hypercube. Initially, the whole repository has been gridded into several hyper-cubes according to the range of objective values. The number of hyper-cubes is a random integer. Those grids that contain no particles have a fitness of zero. Those grids that contain more than one particle are given a fitness value equal to the number of particles in that grid divided by an integer (This integer can be any value, e.g., 5, 6, 20, and any value that decreases the fitness). The fitness values here are the reference used for the roulette wheel selection afterwards. The aim of this division is to decrease the fitness of those grids as a form of fitness sharing [121]. After that the roulette wheel selection algorithm is used to pick the grid from all of those hyper-cubes. The possibility of being selected is on the basis of the fitness values of each grid. The grids with more particles (largest fitness values) have more chances to be selected as the global best grid. At last, one particle *p* from that particular hypercube (global best grid) is randomly selected to be the global best *REP[p]*.

The Pareto algorithm not only focuses on one global optimum in the solution space but the whole trade off response surface, however in this method, both the advantages and disadvantages are prominent because it allows the search algorithm to handle independent multi-objectives. But if too many objectives are to be found the results may be ambiguous and the Pareto optimal sets may also become overwhelming due to the increasing number stored in the repository. Therefore, the selection of the global best particle requires a more advanced technique.

4.5.3 Handling constraints in MOPSO

There are many techniques used to handle constraints for transforming a constrained problem into an unconstrained problem. Gen and Cheng [88] summarised those techniques into four major groups, including a rejecting strategy, a repairing strategy, a modifying strategy, and a penalising strategy. However, Carlson et al. [122]

claimed that the researchers tended to choose only a single method. The most commonly seen strategy used to accommodate the inclusion of violation is the penalty scheme [123]. Instead of totally denying there is an infeasible solution in a problem, the penalty scheme imposes a punishment value on top of the objective function to make that particular solution worse than any other feasible ones. By adding this penalty term the algorithm can have a larger exploration of the solution space than traditional methods.

However, tuning for the suitable penalty function is more or less based on trial and error. An excessively heavy penalty could lead the algorithm to pre-mature, while under penalty may cause the algorithm to wander in the solution space without convergence or take more time than necessary.

4.5.3.1 Penalty strategy

The penalty strategy transfers a constrained problem into an unconstrained problem by penalising the violated solutions. In general, an appropriate penalty function scheme can lead the optimisation to healthy direction, while a poor penalty function scheme drives the search divergence. Setting up a penalty function is according to objective function, because the penalty function should either return a dimensionless value or a value with the same unit of objective function. For example, if one objective of a problem is to minimise stress, the penalty function should give terms that are either dimensionless or in the unit of pressure. The purpose is to remain consistent with the objective functions because the selection of these two can be arbitrary. Another useful hint in building a penalty function is to make the penalty proportional to the constraint violations, such that a heavy penalty is associated with serious constraints, while minor constraints deserve a mild penalty. Thus the essential spirit of applying a penalty is to find suitable penalties. In order to achieve a satisfactory level, Coello [123] proposed that the penalty should be kept as low as possible in its minimum penalty rule.

Penalties are always associated with the extent of the violations that candidates have. Generally speaking, high penalties should be imposed for a serious violation, while low penalties are for small violations. However, if the violation exceeds the constraints to a large extent and/or it is far from reasonable that it cannot achieve

optimal, the death penalty approach may be used to kill off that particular candidate. The death penalty should only be applied very carefully as it indicates that information to the candidate is totally lost. Frequent use of the death penalty does not allow particles to effectively explore the search space. In cases where feasible solutions are rare, this penalty cannot provide any useful information to the PSO.

Penalty functions can be various during search procedures. One of the dynamic penalty pioneers Joines [124] proposed having a lenient penalty around the start of the search and a conservative one in the latter stages. This is called deterministic, where the penalty coefficients are calculated through an iteration number. The benefit of this dynamic penalty is that it allows candidates to explore the infeasible region better, whilst the feasible zone can be located near the end of the search. .

Another dynamic penalty approach proposed by Hinterding [125] utilised feedback from the search to modify the penalty values accordingly. The feedback is based on the ratio of a certain amount of the best feasible solutions to a certain amount of infeasible solutions. If the ratio is too high, the penalty coefficients are increased, but if the ratio is low the penalty coefficients are decreased accordingly. This adaptive approach was used in GA by Davis [126]. However, few references have applied the self-adaptive approach in MOPSO.

4.5.3.2 Repair algorithm

Repair algorithms modify the non-complying candidates that violate constraints to make them feasible, but a specific knowledge of the domain is often required to use the repair mechanism. One of the limitations of this algorithm is that programs cannot always be intelligent enough to make the repair, so the engagement of experienced engineers is vitally important. Moreover this can be very good in design problems when there are experienced experts involved.

4.5.3.3 Strategies in MOPSO

In a single objective PSO, the penalty strategy transfers violations as an extra component in an objective function to increase its value. Therefore, the algorithm accepts the total objective value, which indicates an inferior particle compared to a no penalty particle.

Since there is more than one objective in MOPSO, the penalty strategy can be applied to the relevant objective, or either of the objectives, while in some cases penalty values can be applied on all the objectives.

4.6 Chapter Summary

This chapter has reviewed the concept and mathematical formulation of genetic algorithms (GA), single objective particle swarm optimisation (PSO) and multi-objective particle swarm optimisation (MOPSO). The Matlab global optimisation toolbox was used as the GA engine. GA & PSO Matlab codes employed in this study can be found in Appendix C. The general procedures for applying this optimisation approach to practical problems were listed and the parameters related to this approach were discussed. Furthermore, the advantages and disadvantages of related works of PSO and techniques for improving the performances that drive particles from infeasible to feasible search spaces were presented.

The practical application of GA, PSO and MOPSO to engineering applications involves numerous issues and measurements because information tells us there is no universal technique suitable for every problem. The most popular method for handling constraints is the penalty strategy. Most importantly, this strategy can be applied both in GA, PSO and MOPSO in terms of constraints violation. Other methods can be applied as an auxiliary to improving the searching performance.

5 BENCH MARKING OPTIMISATION ALGORITHMS

5.1 Introduction

Of all the evolutionary techniques introduced in Chapter 2, Sooda et al [127] claimed that the genetic algorithm (GA) performed successfully. Other evolutionary techniques that can be applied to offshore optimisation problems is the particle swarm optimisation technique [107]. In order to find a robust optimisation technique to use in the field of offshore cable design, a benchmark of some stochastic search methods has been carried out.

This chapter compares the performance and efficiency of two evolutionary computational optimisation approaches, particle warm optimisation (PSO) and genetic algorithm (GA). They have been applied to both the mathematical benchmark functions and an engineering application. In the purely mathematical function benchmarking, GA and PSO performed in non-constraint problems, and then the optimised solutions were compared to the theoretical solutions of those mathematical functions. For the engineering application, the problem has been set with constraints and penalty functions.

5.2 Optimisation algorithms in benchmark functions

For the purpose of comparison of GA and PSO, five mathematical benchmark functions were taken from Yao's [128] and Premalatha's [40] papers, and have been summarised in Table 5.1 below. The purpose of this benchmarking is to find a robust technique that is suitable for optimising offshore cable structures. To define the term 'robust' in this chapter, the optimisation technique should at least have the following characteristics.

- The ability to search for the best solution in a problem.
- Good computational efficiency.
- Have a relatively easy operational process that requires less input from professionals.

The efficiency of optimisation methods is also important. The ability to find the global optimum of GA and PSO has been demonstrated by many researchers. Chapter 2 and 4 have details of these two algorithms. In the topic of optimising offshore cable structures, the numerous requirement of simulation time is one of the obstacles that have barred a wider application in this area. Therefore, the appropriate optimisation technique should be efficient in time and have less parameters to manipulate. In other words, it is important to know which technique can find a solution with the smallest population and least iterations.

Table 5.1 Benchmark Functions

Name	**Functions**	**Dimension**	**Initial range of x_i**
F1 De Jong's Function	$f(x)=\sum_{i=1}^{n} x_i^2$	10	±5.12
Rosenbrock's Valley	$f(x)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(1-x_i\right)^2\right]$	10	±2.048
Rastrigin's Function	$f(x)=10n+\sum_{i=1}^{n}\left[x_i^2-10\cos\left(2\pi x_i\right)\right]$	10	±5.12
Schwefel's Function	$f(x)=\sum_{i=1}^{n}\left[-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)\right]$	10	±500
Griewangk's Function	$f(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	10	±600

5.2.1 Benchmark functions

All the benchmark functions are chosen as high dimensional problems. F1 De Jong's function, Rosenbrock's Valley function, and Schwefel's function are uni-modal where there is only one minimum, whereas Rastrigin's function and Schwefel's function are multi-modal. With the increasing of dimensionality, the quantity of local minima increases exponentially. The dimensionality of all the benchmark functions is selected as 10.

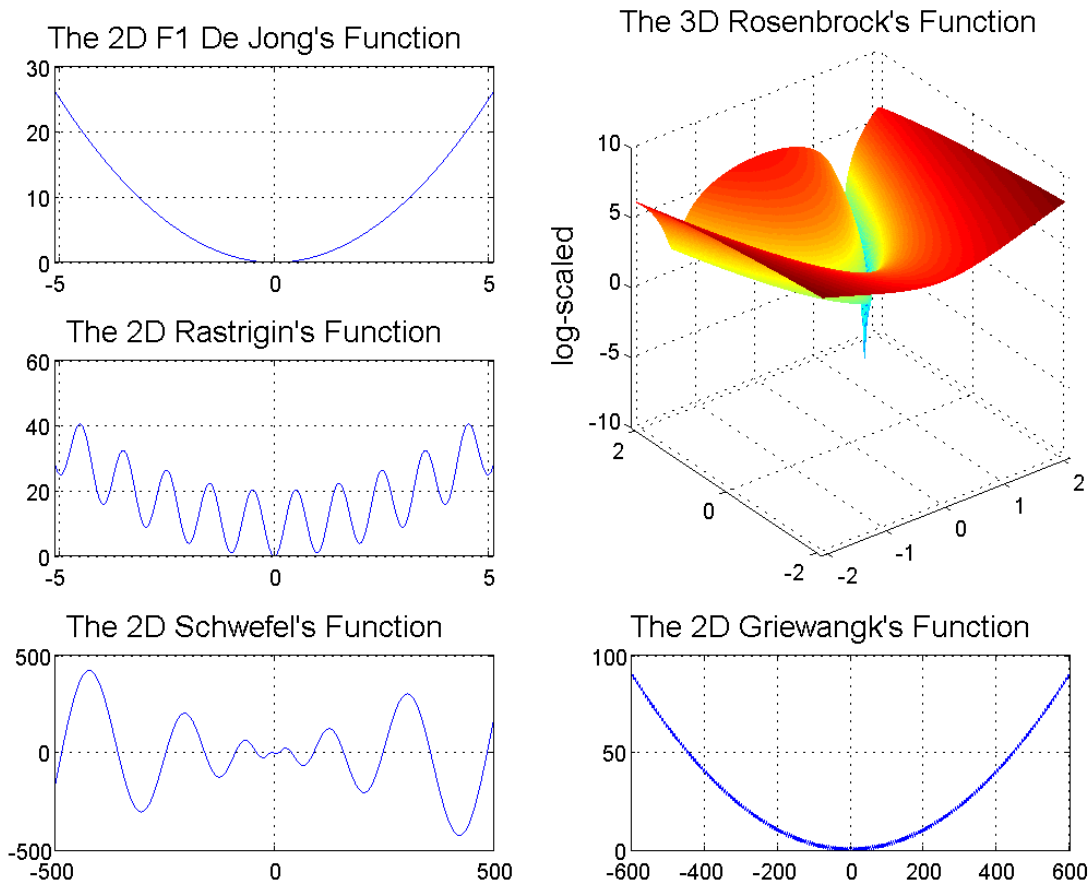

Figure 5.1 Graphs of benchmark functions

Figure 5.1 shows the graph of these five benchmark functions in their smallest dimensional presentation. Except for Rosenbrock's Valley function, the smallest dimension all the remaining benchmark functions is two, so their graphs are presented in the x and y plot. Rosenbrock's valley function has three as its smallest dimension, so for clarity it has been plotted in a log scaled form so the global minimum can be seen. These functions are considered to be good benchmark functions for an optimising program because they include several local minima and only one global minimum. For example, the best minimum in Schwefel's function is geometrically distant from the global minimum and algorithms tend to be trapped in local optima.

The use of high dimensionality in benchmarking is due to the nature of offshore optimisation problems; dimensionality is associated with the number of variables in an optimisation problem. The quantity of independent variables normally has a relationship that is proportional to the complexity of a problem. The greater the

number of design variables a problem has, the harder it is for an algorithm to find its optimum. In the field of optimising offshore cable structures, the number of design variables is less than ten. For example, de Pina [107] used six independent design variables to optimise offshore oil production risers, Boulougouris [112] had eight design variables to optimise a floating LNG terminal, and Cunliffe [129] had seven design variables in an SCR optimisation problem. Therefore, benchmarking these functions in a dimensionality of ten is considered to be appropriate.

5.2.2 GA & PSO parameters and operators

For the purpose of benchmarking these two algorithms employed in this study, details are reported in the optimised coefficients of both algorithms as follows. The coefficients and operators are given below, together with the values in Table 5.2.

Table 5.2 PSO and GA configurable coefficients

	PSO Configuration	**GA Configuration**
Population	20	20
Number of Iterations	100	100
Cognitive coefficient (c_1)	Equation (4.7)	N/A
Social coefficient (c_2)	Equation (4.7)	N/A
Inertia coefficient (w)	0.8	N/A
Selection method	N/A	Tournament
Crossover rate	N/A	0.8
Mutation	N/A	Adaptive feasible
Elite number	N/A	2

It has been proven by Wolpert and Macready [130] that within certain assumptions, there is no one algorithm that is the best for all problems. Furthermore, there is no one set of parameters for a single algorithm that is best for all problems either. For GA and PSO to find the global minimum of those benchmark functions may take more than 100 iterations. Since both approaches has proved to be successful in finding the global minimum for these benchmark functions in Yao's [128] and Premalatha's [40] paper, the main focus of benchmarking these two algorithms is on

computational efficiency. The performances of both GA and PSO are evaluated at the end of 100 iterations.

Eventually, if sufficient iteration numbers were combined with suitable individual sets of parameters for these two algorithms, a near global minimum value can be found. Table 5.3 shows the optimum solutions found by GA and PSO for these five benchmark functions. Since the ability to find the best solution in benchmark functions is not the main aim of this study, computational efficiency is of more interest, and hence the results reported in the 100th iteration are examined in terms of their performance and efficiency.

Experiments were carried out in two groups with respect to initial candidates (population). The initial population of one group was generated uniformly, but on a random basis, in the range given in Table 5.1 for all runs. The other group kept the same initial population for both GA and PSO to avoid the issue of randomness.

Ratnaweera [97] proved that reducing the cognitive acceleration coefficient of each iteration while increasing the social acceleration coefficient would explore the search space widely in a purely mathematical search and have a healthier search experience. Therefore, the acceleration coefficients have been taken from Equation (4.7) for comparative purposes.

5.2.3 Results and comparison of benchmark

Figure 5.2 and Figure 5.3 show the global minima versus iterations from both GA and PSO for all the five given benchmark functions shown graphically in Table 5.1. The fitness values were plotted in the log scaled axis for clarity. The blue dashed lines in these graphs represent the GA's best global candidates and the solid red lines indicate the PSO's global best position in the whole swarm. A combination of final results shown in Table 5.3, and the best global patterns in Figure 5.2 and Figure 5.3, shows that it is easy to conclude that both algorithms are capable of finding the near optimum solutions.

Table 5.3 Benchmark function optimisation results from GA and PSO

Fn Name	Me-thod	*f(x)*	Variables									
			x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
DJ	**GA**	5.02103E-09	-3.9E-05	-7.6E-06	-2.5E-06	1.45E-05	3.73E-05	-1.38E-06	3.36E-05	-2.62E-05	-3.7E-06	1.06E-06
	PSO	5.12703E-13	3.07E-07	-3.72E-07	-9.6E-08	-3.12E-07	1.44E-07	1.93E-07	-2.6E-07	1.36E-07	-1.7E-07	1.64E-08
	Analy	0	0	0	0	0	0	0	0	0	0	0
RV	**GA**	0.000811	0.97239	0.913261	0.849115	1.006635	1.001650	0.863724	1.022321	0.923904	0.889392	0.563994
	PSO	0.02084	0.99941	0.999219	0.998002	0.995507	0.991558	0.983242	0.967305	0.936416	0.877236	0.768571
	Analy	0	1	1	1	1	1	1	1	1	1	1
RF	**GA**	1.30E-06	5.77E-06	2.99E-05	9.14E-06	3.70E-05	-1.1E-05	-8.95E-06	4.03E-05	3.63E-05	-1.6E-05	2.80E-05
	PSO	0.02695	-5E-05	0.000153	0.006409	-0.00375	-0.00131	-0.00013	-0.00011	0.002652	-4.4E-05	-0.00848
	Analy	0	0	0	0	0	0	0	0	0	0	0
SF	**GA**	-4189.829	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687
	PSO	-4189.829	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687
	Analy	-4189.829	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687
GF	**GA**	0.000316	0.004306	0.018999	0.003	0.003442	0.027731	0.027086	0.023527	0.017618	0.016987	0.000949
	PSO	1.92E-13	2.35E-07	-3.86E-07	1.02E-06	-5.71E-07	1.09E-06	4.87E-07	8.44E-07	-8.05E-08	-3.5E-07	1.92E-08
	Analy	0	0	0	0	0	0	0	0	0	0	0

Note: DJ, F1 De Jong's Function; RV, Rosenbrock's Valley Function, RF, Rastrigin's Function; GF, Griewangk's Function. Analy; functions' analytical minimum.

Table 5.4 Benchmark function results comparison with random initial candidates

Fn Name	Me-thod	*f(x)*	Variables									
			x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
DJ	**GA**	0.002752	0.022026	0.035852	-0.0056	-0.00473	-0.00648	0.015398	0.001306	-0.00991	0.022723	0.005702
	PSO	0.003808	0.006663	0.008146	-0.01341	0.019537	-0.02848	0.014454	-0.00245	-0.0306	-0.0185	-0.02884
	Analy	0	0	0	0	0	0	0	0	0	0	0
RV	**GA**	0.183782	0.989858	1.00736	1.002535	0.991605	0.980749	0.970574	0.954449	0.922452	0.844496	0.71227
	PSO	1.562608	0.992501	0.988137	0.98474	0.965107	0.896334	0.776144	0.640092	0.425128	0.192532	0.039914
	Analy	0	1	1	1	1	1	1	1	1	1	1
RF	**GA**	2.13972	0.021454	-0.99124	-0.00035	0.000724	0.007826	0.002256	-0.00111	1.004701	0.008282	0.007072
	PSO	15.35256	-0.96928	0.990299	-1.0228	0.93264	-1.01144	0.012264	-0.00458	-1.98231	-1.02656	-1.96968
	Analy	0	0	0	0	0	0	0	0	0	0	0
SF	**GA**	-3306.75	-489.513	-277.859	459.187	409.1342	424.5183	-294.442	409.2465	423.1055	420.2371	423.2134
	PSO	-4026.08	426.1113	422.9728	415.4945	421.4630	-313.455	417.4509	420.1107	429.7586	430.3120	422.1470
	Analy	-4189.829	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687
GF	**GA**	2.06959	36.662	20.2078	-9.0299	19.7264	9.2536	-9.745	20.3043	-31.6017	13.4028	-16.8481
	PSO	0.88369	-8.80729	-1.25526	-8.78106	0.52748	7.181025	1.099571	0.564624	2.783809	-1.70192	-7.48312
	Analy	0	0	0	0	0	0	0	0	0	0	0
Note: DJ, F1 De Jong's Function; RV, Rosenbrock's Valley Function, RF, Rastrigin's Function; GF, Griewangk's Function. Analy; functions' analytical minimum.												

Table 5.5 Benchmark function results comparison with the same initial candidates

Fn Name	Me-thod	*f(x)*	Variables									
			x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
DJ	**GA**	0.001699	0.008429	-0.01392	0.004693	0.004322	0.008005	-0.02223	0.014916	-0.01328	-0.02088	0.000023
	PSO	0.01013	0.043709	-0.0363	-0.05775	0.04022	0.001285	0.021876	0.006918	0.007255	-0.00734	-0.03627
	Analy	0	0	0	0	0	0	0	0	0	0	0
RV	**GA**	1.32283	0.990379	0.980376	0.969381	0.940484	0.888687	0.788604	0.617933	0.383442	0.150443	0.026062
	PSO	6.62202	0.751846	0.546456	0.310197	0.105409	0.02242	0.005274	-0.01101	0.007974	0.021963	0.018382
	Analy	0	1	1	1	1	1	1	1	1	1	1
RF	**GA**	1.34234	0.006482	0.001541	0.035405	0.010086	0.010379	-0.00105	0.007874	0.006742	0.007449	1.00417
	PSO	18.93033	-1.01739	-2.00548	-0.08852	0.028676	0.871697	-1.01444	0.02132	-0.14073	0.997459	-0.11364
	Analy	0	0	0	0	0	0	0	0	0	0	0
SF	**GA**	-3045.19	-298.849	-142.741	-299.917	420.4999	429.7488	438.5725	-297.189	423.9826	-24.5423	416.802
	PSO	-3566.96782	-500	423.2184	424.4651	421.7073	423.7772	421.6230	431.4794	429.4731	-500	-299.328
	Analy	-4189.829	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687	420.9687
GF	**GA**	14.525	-32.6644	108.4644	0.26274	-70.763	-82.1848	93.03749	34.69951	46.51711	-69.2265	-112.731
	PSO	0.61763	-0.25702	-0.99161	-0.59352	2.249363	-9.77078	0.193653	1.803794	18.41743	-0.11506	-9.15899
	Analy	0	0	0	0	0	0	0	0	0	0	0

Note: DJ, F1 De Jong's Function; RV, Rosenbrock's Valley Function, RF, Rastrigin's Function; GF, Griewangk's Function. Analy; functions' analytical minimum.

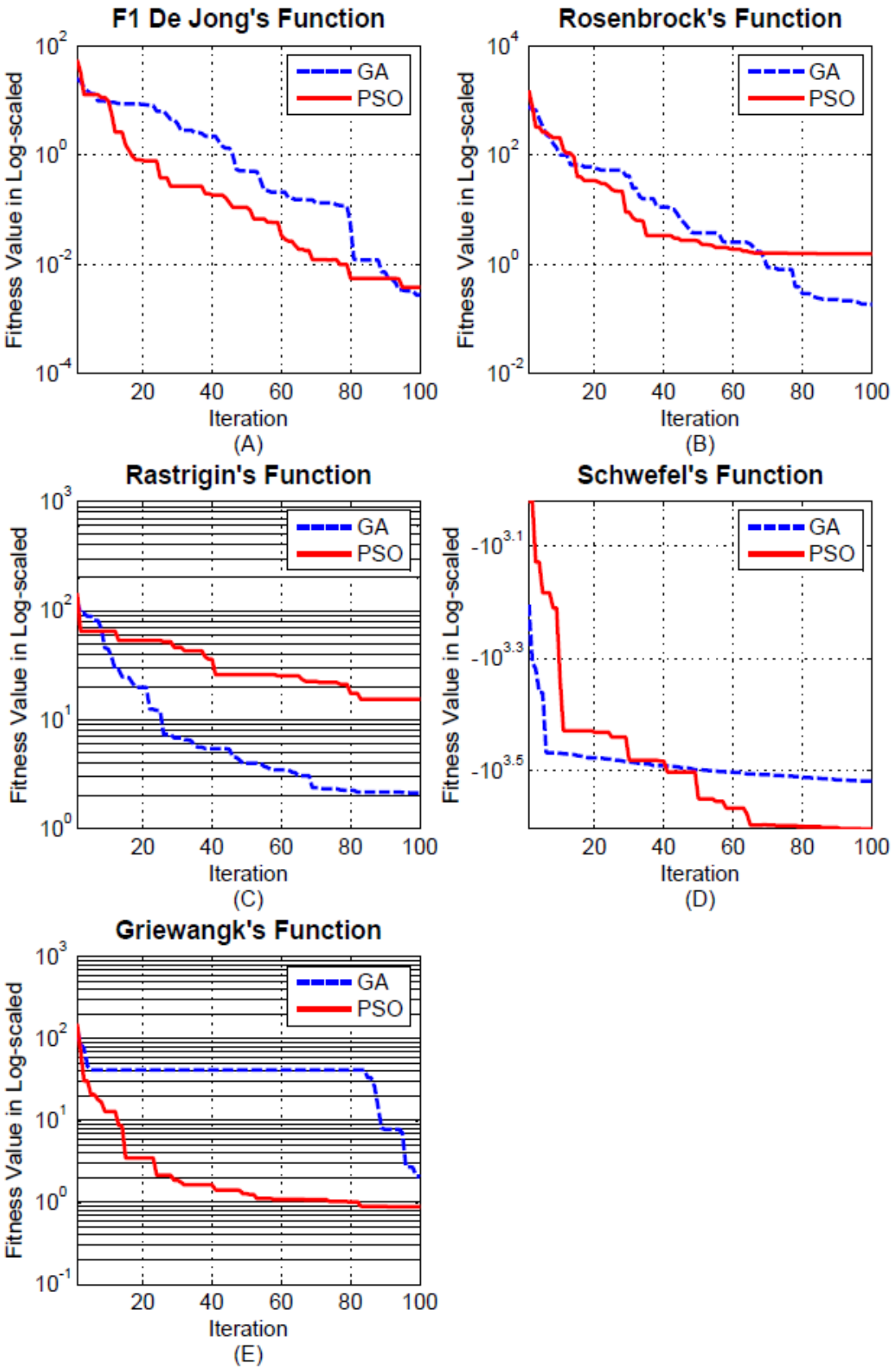

Figure 5.2 GA & PSO performance comparison with **random** initial candidates for 5 benchmark functions

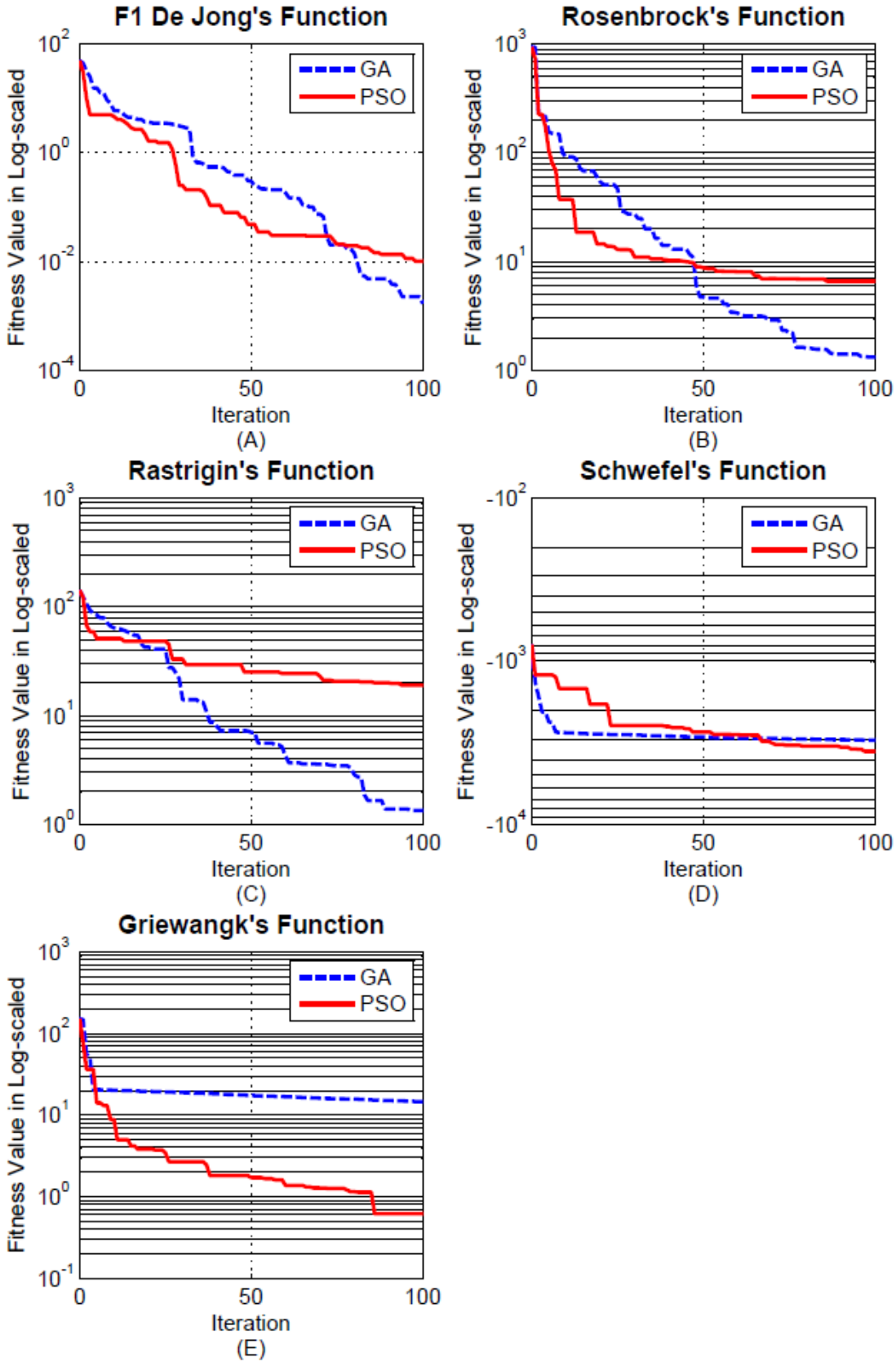

Figure 5.3 GA & PSO performance comparison with the **same** initial candidates for 5 benchmark functions

Except for Schwefel's function, all the remaining functions have a theoretical minimum value of zero. Schwefel's function has the smallest global value of -4189.829 in 10 dimensions. F1 De Jong's function, Rastrigin's function, and Griewangk's function have the same identical x_i value of zero to get their minimum, while Rosenbrock's Valley function reaches the global minimum for all the x_i that are equal to one. Schwefel's function has its smallest value of -4189.829 when all the x_i are equal to 420.9687.

Optimised results from GA in F1 De Jong's Function, Rosenbrock's Valley, and Rastrigin's Function are better at the end of the 100th iteration for both groups, but the PSO had a smaller fitness value in Schwefel's function and Griewangk's function. After closely examining the performance curve of F1 De Jong's function (see (A) in Figure 5.2), it is arguable that if the GA has a better solution it may be due to the better quality of initial population. The same experiments were therefore carried out with the same initial population of both GA and PSO. In Figure 5.2, the iteration starts from No.1 while in Figure 5.3, it starts from No. 0. This is because of the ignorance of the 'parent' when plotting the performance curve in Figure 5.2.

Table 5.4 summarised the theoretical results for all of the five benchmark functions in comparison to both the PSO and GA where the initial populations were randomly generated. The results of the GA and PSO were taken from the best of 10 runs. Table 5.5 gave the results of the comparison within the same initial population generated for both GA and PSO in the specific range defined in Table 5.1. The best of all five benchmark functions given means that their minimum fitness values were from a total of ten runs. The true minimum of these functions are also given for reference.

5.2.4 Benchmark results discussion

From the results above it is difficult to conclude which optimisation algorithm is better than the other. For some benchmark functions such as the F1 De Jong's function, Rosenbrock's Valley and Rastrigin's, the GA had better solutions within a certain amount of computational runs, whereas the PSO showed a great efficiency with Griewangk's and Schwefel's function.

Even though the parameters of GA and PSO used in these benchmark functions were optimised for the purpose, it is possible that there is no universal set of parameters that fit every function. Different problems have different favourable sets of parameters. Sometimes the same problem would have different favourable sets of parameters in dissimilar situations. Therefore, with the performance plots of GA and PSO for the benchmark function alone, it is disputable to conclude in terms of the computational efficiency of these two algorithms.

Table 5.6 GA & PSO results comparison based on 10 runs

Fn Name	UGIP (Group 1)			SIP (Group 2)		
	Method	**Mean**	**Std. Dev**	**Method**	**Mean**	**Std. Dev**
DJ	**GA**	0.0340	0.0290	**GA**	0.0263	0.0329
	PSO	0.0239	0.0226	**PSO**	0.0335	0.0217
RV	**GA**	32.52	33.80	**GA**	45.49	44.12
	PSO	10.23	3.89	**PSO**	9.10	1.77
RF	**GA**	7.59	4.48	**GA**	7.76	5.03
	PSO	25.81	8.21	**PSO**	27.14	5.52
SF	**GA**	-2623.8	421.0	**GA**	-2478.3	374.8
	PSO	-3352.2	327.7	**PSO**	-3228.8	259.1
GF	**GA**	5.74	1.83	**GA**	40.91	23.11
	PSO	1.06	0.14	**PSO**	1.18	0.34

Note: DJ, F1 De Jong's Function; RV, Rosenbrock's Valley Function, RF, Rastrigin's Function; GF, Griewangk's Function; UGIP, Uniformly Generated Initial Population group, SIP, the Same Initial Population group.

The performance curves in Figure 5.2 and Figure 5.3 gave the best out of a total of ten runs. Taking the average performance and efficiency into consideration, Table 5.6 has been summarised with the mean and standard deviation values of the ten runs of each benchmark function. The winning components in the table have been shaded. The lower mean values means that the specific algorithm gives a generally better solution in ten runs. In other words, within the same amount of calculation loads the lower mean value represents a better efficiency. The lower values of the standard deviation indicate that the specific algorithm has a smaller variance in ten runs. Thus, a more reliable performance is expected from that particular algorithm.

In the different initial population (Group 1) group, the PSO showed a better mean and standard deviation in four out of five benchmark functions. The GA dominated these two items in Rastrigin's function, while in the group of the same initial population (Group 2), except for the F1 De Jong function where GA has a better mean but PSO has a better standard deviation, the other four benchmark functions shared the same results as group 1. In terms of reliability, the PSO seems to perform better in Group1 and 2, with the exception of Rastrigin's function. To sum up, it can be concluded that both optimisation algorithms are capable of finding near global optima solutions. However, it is insufficient to distinguish which algorithm is superior to the other within these benchmark functions alone.

5.3 Benchmark in 10-bar truss classical problem

Since both GA's and PSO are evolutionary global solution search algorithms and it is difficult to judge their performance in purely mathematical functions, it would be of interest to have their effectiveness compared on the same structural optimisation problem. The 10-bar redundant truss optimisation problem (shown in Figure 5.4) was selected for this purpose. The structural optimisation problem selected herein is a classical problem in optimisation field. Many researchers such as Galante [131], Jingui et al [132], Rajeev et al [133] and Leite et al [134] published their works about this problem. The optimised results have reached a consensus solution.

5.3.1 10-bar truss redundant problem

The 10-bar truss redundant problem may be solved as a continuous problem, which assumed that all the members of the truss have a continuous range of selection of the cross section. The problem has been solved as a continuous problem in both Perez and Sunar's works [93, 135], where the area of the cross section varies between 0.1 $in.^2$ (64.516 mm^2) and 35.0 $in.^2$ (22580.6 mm^2). As shown in Figure 5.4, all the solid circles are the numbers of the truss nodes, and the hollow circles located in the middle of truss bar are the number of members.

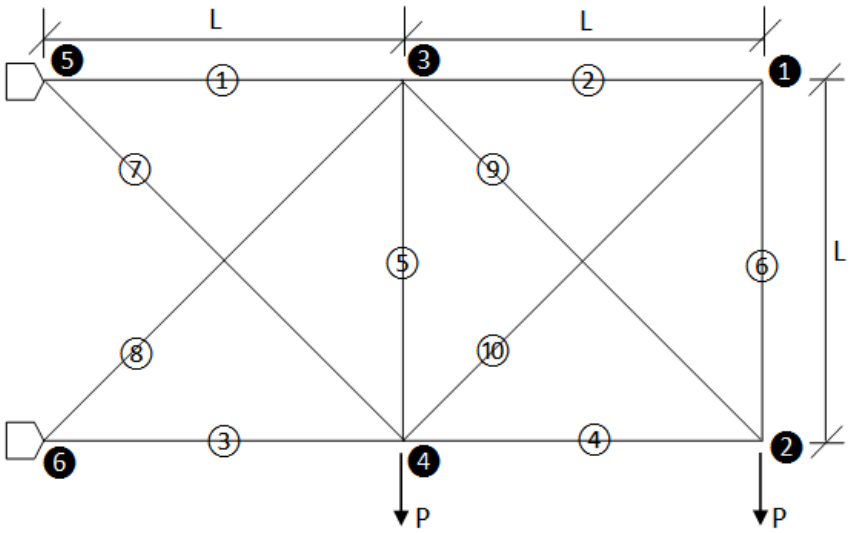

Figure 5.4 10-bar redundant truss

In this problem the truss is to be optimised for a minimum weight, given that the density of the material is 0.1 lb/in^3 (2767.9905kg/m^3), and Young's modulus is 104 ksi (68.9476 GPa). The allowable stress for every member cannot exceed 25 ksi (172.3689 MPa), and the maximum deflection of any nodes in both directions is ±2 inch (50.8mm). The original problem was solved with imperial units. The current work was done in SI units. Therefore, the material properties and constraints have been taken in SI units in a rational format as:

Table 5.7 Material properties and constraints

Young's modulus (E)	68.9GPa
Material density (ρ)	2770kg/m3
Maximum allowable stress (σ)	172MPa
Maximum allowable displacements (δ)	50.8mm

The reason for selecting the 10-bar truss redundant problem as the benchmark is not its complexity, it is actually relatively simple. However, even though it is a simple optimisation problem, it cannot easily be solved by conventional searching methods, which indicates just how these approaches to optimisation work, and also demonstrate their performance and effectiveness.

It is not realistic to treat this truss problem as continuous since most steel truss members are made to the predetermined sizes provided in the form of a manufacturer's handbook. This means it is necessary to solve this optimisation problem as a discrete problem, apart from the obvious reason that there is a potential benefit to solving this problem in a discrete mode. References associated with the PSO [80, 93, 136] algorithm use continuous structural applications. It is worthwhile exploring the performance and effectiveness of PSO in the range of discrete problems, so all member sections have been chosen from the American Institute of Steel Construction (AISC) Manual. All of the sections have been tabulated below [19].

Table 5.8 AISC truss member section table

Section ID	**Area (in.2)**	**Area (mm2)**	**Section ID**	**Area (in.2)**	**Area (mm2)**	**Section ID**	**Area (in.2)**	**Area (mm2)**
1	1.62	1045	**15**	3.63	2342	**29**	11.50	7419
2	1.80	1161	**16**	3.84	2477	**30**	13.50	8710
3	1.99	1284	**17**	3.87	2497	**31**	13.90	8968
4	2.13	1374	**18**	3.88	2503	**32**	14.20	9161
5	2.38	1535	**19**	4.18	2697	**33**	15.50	10000
6	2.62	1690	**20**	4.22	2723	**34**	16.00	10323
7	2.63	1697	**21**	4.49	2897	**35**	16.90	10903
8	2.88	1858	**22**	4.59	2961	**36**	18.80	12129
9	2.93	1890	**23**	4.80	3097	**37**	19.90	12839
10	3.09	1994	**24**	4.97	3206	**38**	22.00	14194
11	3.13	2019	**25**	5.12	3303	**39**	22.90	14774
12	3.38	2181	**26**	5.74	3703	**40**	26.50	17097
13	3.47	2239	**27**	7.22	4658	**41**	30.00	19355
14	3.55	2290	**28**	7.97	5142	**42**	33.50	21613

Transferring this continuous problem to a discrete problem for this particular truss example is simple. The conservative method is to round each member up to the next largest member from the AISC table, or round down to the previous largest member. The conservative approach is adopted here.

5.3.2 Objective and definition of the penalty functions

The total mass of the truss plus a penalty mass can be developed to achieve the goal of finding the minimum weight subject to stress and displacement constraints. For this particular problem the penalty function has been set to allow for constraint violations, and it was decided not to punish any members that did not violate these constraints. The following objective and penalty function has been set after experiments.

$$F_{obj} = \sum_{i=1}^{6} \rho A_i L + \sum_{i=1}^{4} \rho A_i \sqrt{2} L + P \tag{5.1}$$

$$P = \sum_{i=1}^{10} \epsilon_i \left(150^{(\sigma_i/\sigma)}\right) + \sum_{i=1}^{8} \epsilon_i \left(270^{(\delta_i/\delta)}\right) \tag{5.2}$$

where σ_i is the stress in a particular truss member, δ_i is the deflection of a node in a particular direction, σ and δ is the maximum allowable stress and displacements respectively referred to in Table 5.7, ϵ is the judgement coefficient where $\epsilon = 1$ when $\sigma_i > \sigma$ or $\delta_i > \delta$, $\epsilon = 0$ when $\sigma_i < \sigma$ or $\delta_i < \delta$. Equation (5.2) is the penalty function used here to impose extra mass if any violations have been detected. Compared to the stress penalty, a slight heavier mass penalty is used here, because the deflection violations in this problem are sensitive. The penalty threshold of Equation (5.2) has been set to assure the non-violation compliance at the final answers.

5.3.3 Parameters and experimental configuration

For this proposed problem the PSO cognitive acceleration coefficient c_1 and social acceleration coefficient c_2 are assigned as 2, the inertia coefficient w is 0.8. Instead of using dynamic coefficients, the static constants of these coefficients were used for the purpose of simplifying the PSO. GA has a tournament selection algorithm with an elite number of 2.

To examine the efficiency of these two algorithms, the candidate number was tested on the basis of 10, 50, 100, 150, and 200. The maximum number of iterations was selected from 100 to 800 times of runs at 100, 200, 400, and 800 termination criteria respectively. Therefore, experiments were organised in groups of two in terms of those two coefficients, as shown in Table 5.9.

Table 5.9 PSO and GA coefficients

	PSO Configuration	**GA Configuration**
Population	10,50,100,150,200	10,50,100,150,200
Number of Iterations	100,200,400,800	100,200,400,800
Cognitive coefficient (c_1)	2	N/A
Social coefficient (c_2)	2	N/A
Inertia coefficient (w)	0.8	N/A
Selection method	N/A	Tournament
Crossover rate	N/A	0.8,
Mutation	N/A	Uniform
Elite number	N/A	2

5.3.4 Results from references

For this classical truss problem there were a number of literatures [41, 131, 137, 138] claiming that they found the optimum solution, and it was also believed that this problem was optimised to a minimum weight of 2490.6kg. Some of the references have been summarised and the results have been tabulated in Table 5.10. It is noted that the results in the references may differ from the table here due to the unit conversion and rounding errors. Moreover, solutions that comply in the references may also have violations as the properties and constraints that were used for analysis are from Table 5.7 with SI units. These values contain rounding errors due to the conversion of the two units system. Even though those violations after units conversion are negligible, it may still affect the search behaviour of algorithms when trying to reproduce these solutions.

The best solutions obtained from GA and PSO are summarised in Table 5.10. First of all, both the GA and PSO can find a near optimum solution to the 10-bar redundant truss problem. Second, the PSO has a slightly better configuration than the GA in the particular sets of coefficients pre-settled. It should be noted that for a comparison between the results of this study to literature, the unit conversion rounding errors was not taken into account. The comparison between GA and PSO is under the SI unit system.

Table 5.10 Comparison of optimum solutions

	Weight	***δ***	**Truss member ID**									
	(kg)	**(mm)**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
I.	2546.4	-50.5	42	1	38	33	1	1	32	37	37	6
II.	2475.9	-51.1	42	1	38	32	1	1	28	39	38	1
III.	2490.6	-50.5	42	1	39	32	1	1	28	39	38	1
IV.	2490.6	-50.5	42	1	39	32	1	1	28	39	38	1
V.	2503.1		42	1	40	33	1	1	28	38	37	1
VI.	2498.7		42	2	39	33	1	2	28	38	38	1

Notes:

I. Rajeev and Krishnamoorthy [137]
II. Galante [131]
III. Cai [138]
IV. Schmidt and Thierauf [41]
V. The best results from GA
VI. The best results from PSO

5.3.5 Results and comparison of benchmark

The tool used here to analyse the 10-bar truss problem was developed by the author in Matlab. Each approach had been chosen to run 10 times, and the best of ten was selected to be the solution shown here from Table 5.11 to Table 5.14.

All the experiments were carried out with a standard desktop at the University of Wollongong. It configured an Intel Core i7 CPU at 2.93GHz and 4GB of RAM with a Windows XP operating system. The averages fitness values of the ten runs were also given as a comparison index. The tabulated results reveal that both PSO and GA are capable of finding feasible solution without any violations in this classical truss problem. A further comparison of these two methods shows that the PSO generally has better results than GA in both the best solution and average solution. Figure 5.5 plots the average weight results for both GA and PSO in an increasing number of candidates. The weight is normalised to the best solution claimed by the reference. The solid line group represents the performance curve of PSO, while the dashed line group shows that of GA. It is clear that all the solid lines are underneath the dashed lines in all the candidate numbers.

Table 5.11 GA and PSO results comparison for maximum 100 iteration runs

	No.C	B.W. (kg)	A.W. (kg)	P (kg)	M.T (s)	Truss member ID									
						1	2	3	4	5	6	7	8	9	10
GA	10	2593.5	2929.9	0	5.93	40	2	39	33	3	6	32	39	39	2
PSO	10	2552.4	2775.6	0	3.37	41	5	38	32	2	4	28	39	40	2
GA	50	2544.7	2770	0	23.92	41	1	41	31	2	1	29	38	36	3
PSO	50	2553.1	2644.4	0	12.48	41	18	40	34	2	2	27	39	28	2
GA	100	2529.9	2615.8	0	48.01	42	1	39	29	1	1	29	38	39	1
PSO	100	2521.9	2569.1	0	25.77	41	1	40	31	1	2	29	37	38	5
GA	150	2511.9	2576.8	0	72.83	41	1	39	31	1	1	29	38	39	1
PSO	150	2512.7	2545.4	0	37.82	41	2	40	34	1	2	28	38	38	2
GA	200	2541.1	2583	0	96.54	41	1	40	30	1	1	30	38	37	1
PSO	200	2507.8	2531	0	46.00	42	2	38	33	2	2	28	39	38	1

Note: No.C, number of candidates; B.W., best weight, A.W., average weight, P, penalty, M.T., mean time

Table 5.12 GA and PSO results comparison for maximum 200 iteration runs

	No.C	B.W. (kg)	A.W. (kg)	P (kg)	M.T (s)	Truss member ID									
						1	2	3	4	5	6	7	8	9	10
GA	10	2578.5	2811.3	0	12	40	7	40	36	1	4	30	38	36	3
PSO	10	2526.9	2697.8	0	6.42	41	4	38	30	1	2	29	39	39	2
GA	50	2616.3	2720.9	0	47.11	40	1	40	34	1	8	35	37	36	21
PSO	50	2515.8	2545.4	0	25.05	41	2	40	31	2	1	28	39	39	1
GA	100	2522.1	2620.1	0	94.97	42	1	40	32	1	2	29	37	37	2
PSO	100	2503.1	2519.0	0	43.60	41	2	39	35	1	1	28	39	39	1
GA	150	2523.3	2593.8	0	141.1	41	1	38	33	1	1	29	38	39	1
PSO	150	2500.0	2515.8	0	74.48	42	2	39	33	1	1	28	38	38	2
GA	200	2518.6	2569.5	0	185.9	42	1	40	31	1	1	29	37	37	3
PSO	200	2502.2	2509.9	0	99.77	42	2	39	33	1	2	27	39	38	1

Note: No.C, number of candidates; B.W., best weight, A.W., average weight, P, penalty, M.T., mean time

Table 5.13 GA and PSO results comparison for maximum 400 iteration runs

	No.C	B.W. (kg)	A.W. (kg)	P (kg)	M.T (s)	Truss member ID									
						1	2	3	4	5	6	7	8	9	10
GA	10	2553.3	2920.8	0	23.28	41	1	40	30	1	2	31	38	37	1
PSO	10	2507.2	2647.8	0	12.81	41	2	39	35	1	1	28	39	39	2
GA	50	2577.0	2655.8	0	80.04	40	1	42	30	1	1	30	36	38	2
PSO	50	2512.9	2517.1	0	47.07	41	3	40	34	1	1	28	38	38	2
GA	100	2524.0	2592.5	0	155.1	42	4	39	31	1	6	28	39	38	4
PSO	100	2502.9	2513.0	0	94.54	42	2	39	33	1	2	28	38	38	2
GA	150	2527.2	2585.7	0	248.1	42	1	37	31	1	2	29	39	38	2
PSO	150	2499.5	2511.3	0	139.7	42	1	39	32	1	2	28	39	38	2
GA	200	2510.7	2562.7	0	331.2	41	1	38	34	1	1	29	38	38	1
PSO	200	2498.7	2505.3	0	185.9	42	2	39	33	1	2	28	38	38	1

Note: No.C, number of candidates; B.W., best weight, A.W., average weight, P, penalty, M.T., mean time

Table 5.14 GA and PSO results comparison for maximum 800 iteration runs

	No.C	B.W. (kg)	A.W. (kg)	P (kg)	M.T (s)	Truss member ID									
						1	2	3	4	5	6	7	8	9	10
GA	10	2587.9	2766.3	0	46.08	41	4	38	29	1	4	33	39	39	1
PSO	10	2507.2	2572.1	0	25.66	41	2	39	35	1	1	28	39	39	2
GA	50	2529.9	2715.6	0	121.3	41	1	39	29	1	1	29	38	39	1
PSO	50	2502.4	2509.4	0	99.42	42	2	39	32	1	2	28	39	38	2
GA	100	2527.8	2597.3	0	435.0	41	1	39	30	1	1	28	39	40	1
PSO	100	2500.0	2505.4	0	193.0	42	1	39	32	1	1	27	39	39	2
GA	150	2503.1	2561.9	0	452.4	42	1	40	33	1	1	28	38	37	1
PSO	150	2501.9	2505.9	0	293.6	42	1	38	33	2	1	28	39	38	1
GA	200	2535.4	2578.3	0	563.3	42	1	38	31	1	1	30	38	37	2
PSO	200	2501.9	2507.3	0	384.8	42	1	38	33	1	2	28	39	38	1

Note: No.C, number of candidates; B.W., best weight, A.W., average weight, P, penalty, M.T., mean time

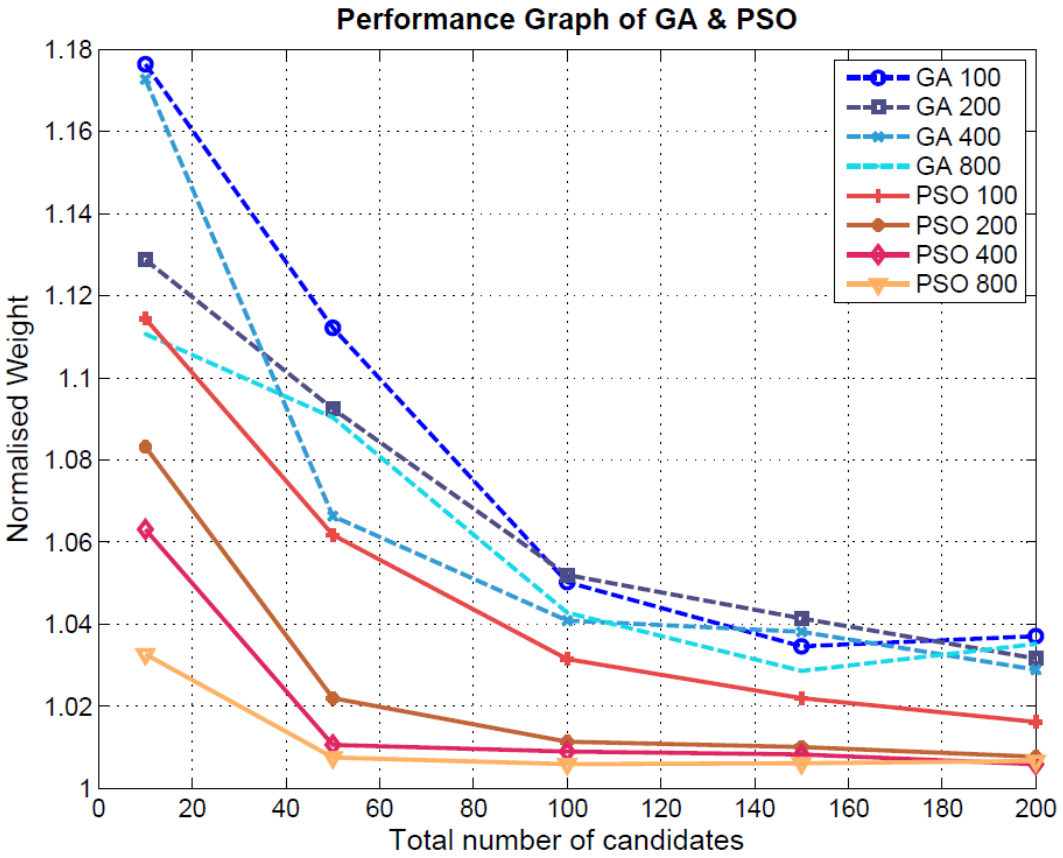

Figure 5.5 Optimisation results comparison for GA and PSO

Table 5.15 PSO and GA average results comparison in quantity manner

Number of Candidates	Maximum iteration							
	100		200		400		800	
	Weight (kg)	Time (s)	Weight (kg)	Time (s)	Weight (kg)	Time (s)	Weight (kg)	Time (s)
10	154.3	2.56	113.5	5.58	272.2	10.47	194.2	20.42
50	125.6	11.44	175.5	22.06	138.7	32.97	206.2	21.88
100	52.9	22.24	101.1	51.37	79.5	60.56	91.9	242
150	23.5	35.01	78	66.62	74.4	108.4	56	158.8
200	49.8	50.54	59.6	86.13	57.4	145.3	71	178.5
Note: positive values in this table mean that PSO better than GA by …								

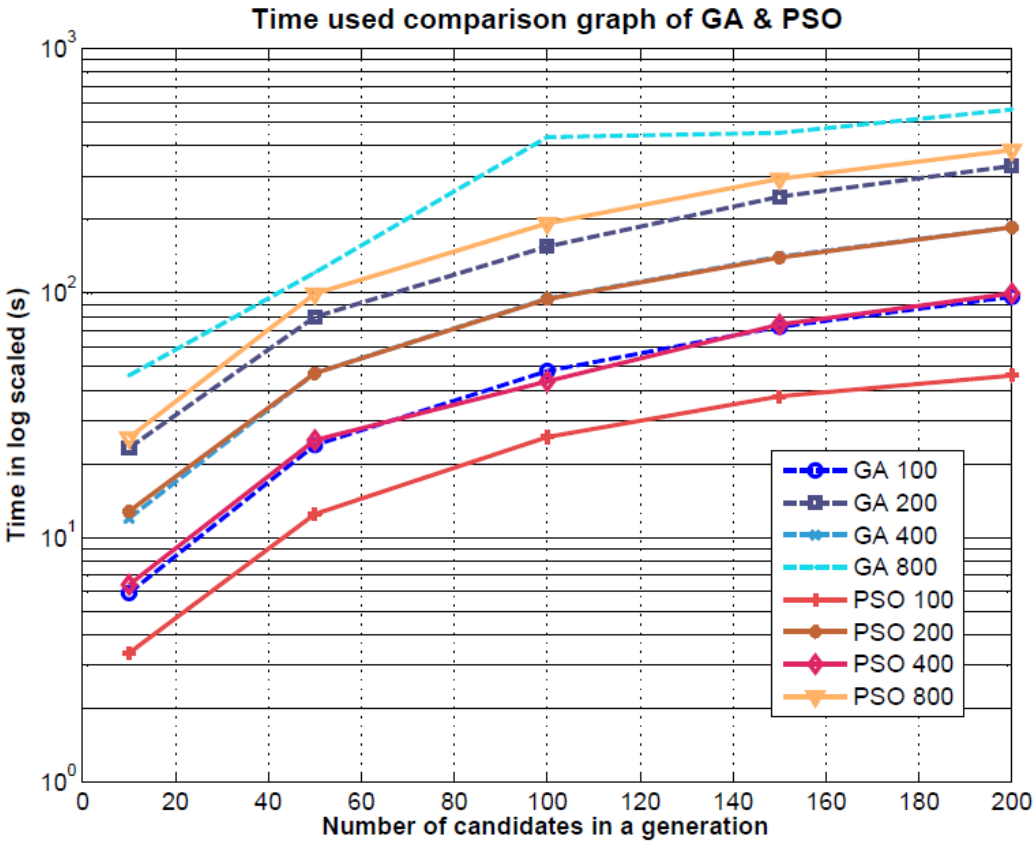

Figure 5.6 Time used comparison for GA and PSO

In order to have a clear view of the results of these two approaches to artificial intelligence, the quantity comparison results of PSO and GA have been tabulated in Table 5.15. The positive values means that PSO is saving when compared with the GA.

Figure 5.6 shows the computational performance of GA and PSO in terms of time consumed. The PSO generally saved time within the same amount of computational loads. Since the method used to calculate the stress and strain of the truss is the same, the time difference showed in the diagraph can be considered as the discrepancy of time required due to the algorithms themselves.

5.3.6 Discussion of the Comparison

On the basis of section 5.3.5, including all the tangible tables and graphs shown here, the following results can be drawn. In all, the PSO generally performed better (average weight) at optimising the truss problem. As seen from Figure 5.5, the PSO group (solid lines) are located underneath the GA. In the graph represented here, the

lower the weight, the better the performance. Figure 5.7 shows the difference in performance when comparing the PSO and GA in terms of their final solution. From the comparison, following conclusions can be drawn:

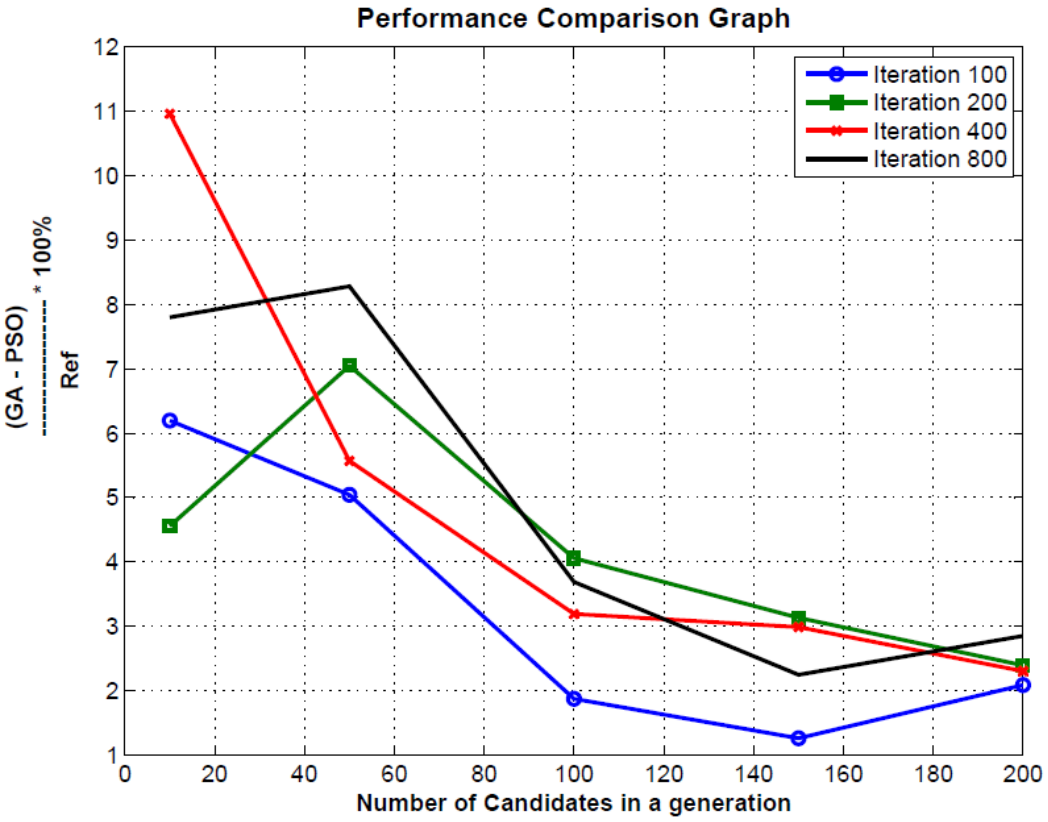

Figure 5.7 Performance comparison graph of PSO and GA

- GA is more sensitive to the number of candidates (population size) than PSO, as the slopes of the GA curves are steeper than the PSO (for 200, 400 and 800 iteration curves). A general descending trend indicates, from the performance comparison graph in Figure 5.7, the performance difference between GA with a larger number of candidates is smaller than PSO.
- PSO is more sensitive to the maximum number of run iterations because the gap between 100 and 800 iterations in Figure 5.5 (solution envelope of PSO) is wider than the GA. This conclusion is not very obvious in that diagram. Table 5.15 explains this conclusion in the weight differences between 100 and 800 iteration termination criteria. Within the same candidate number, the weight differences of the two approaches are larger in the 800 iteration

stopping criteria than that of 100. Graphically, as shown in Figure 5.7, the black curve is generally above the blue one, and therefore the PSO generally performs more reliably with a relatively large number of iterations.

- Running the PSO requires less time than the GA under the same problem. This can be shown in Table 5.15 where the time difference between PSO and GA shows that running PSO within the same calculation load is less.
- Within the same amount of computational loads (the same total number of candidates), the PSO can give better results than GA. This can be fully proven in Figure 5.5.

In an analysis of the problem in terms of the time issue, the PSO has almost halved the time required for GA. There is a potential reason due to the elitism operator used in the GA. The algorithm has to wait until all the population in a generation are completely simulated in order to get the best ones, and in the meantime, some units of a processor may stay idle, which effectively diminishes the GA's time performance. Furthermore, running eight times the amount of case numbers takes roughly eight times longer for PSO. The time required for simulating is strictly linearly proportional to the case to be analysed.

In terms of the quality of the final solution, PSO has demonstrated a superior result under the same computational loads, as shown in Figure 5.7. This feature is particularly important because in offshore optimisation, the use of a simulation package such as OrcaFlex is often considered to be time consuming. Optimisation always involves the evaluation of hundreds and thousands of candidates, so even a single simulation only consumes a few seconds, but the time required for a large number of candidate cases is tremendous. Hence, within a limited amount of candidates, the algorithm that gives relatively superior solutions is computationally time economic, which can be considered as more suitable to apply in the field of offshore optimisation.

Another obvious conclusion that can be drawn from Figure 5.5 is the reliability of these two approaches. All the solid lines (PSO results curve) showed an expected behaviour with the increasing number of candidates and termination number whereas

the dashed lines (GA results curve) were less predictable than the PSO, even though they have a generous trend towards obedience. For example, the GA within the 50 population size and 800 iterations has an inferior performance than the 400 iterations termination with the same population number. This can be explained by the difference in quality between the initial candidates. However, the same challenge in the PSO did not seem to be a problem.

5.4 Selection of search algorithm for offshore marine cable design and optimisation

The selection of PSO as the tool for the optimisation scheme was not random. The processes of reviewing the optimisation techniques in Chapter 4, benchmarked GA and PSO in the different scenarios covered in section 5.2 and 5.3 fully demonstrates the capability of PSO. In summary, the reasons why PSO can be the algorithm for offshore marine cable design and optimisation are as follows:

- First of all, classical methods suffer from fundamental limitations, as they rely on the initial feasible design presented by engineers and then the sequential improvement of such techniques can be made.
- Furthermore, classical approaches are not sufficient enough to handle non-smooth response surfaces or problems with discontinuity. If feasible design solutions cannot be provided by users, or sometimes engineers are expecting the algorithm to provide feasible answers, such methods are generally not reliable or suitable. However, this group of optimisation techniques can be played as a supplementary or secondary approach when near optimum solutions have been found. An improved search can be expected with a good solution.
- Of the artificial intelligence optimisation group, the PSO has been demonstrated as having great advantages in the population size requirement and speed of convergence. Within the same computational loads, it has superior performance and more reliable behaviour.
- Last but not least, the PSO algorithm has relatively less coefficients to manipulate. Developing PSO scripts in a computer manner is relatively simpler than with other approaches.

5.5 Chapter summary

GA, which is a stochastic approach that makes use of natural evolutionary theory, has been shown to be capable of solving both mathematical and engineering applications. Likewise, PSO, which is also a stochastic method use of information shared by the whole swarm, has been proven to be effective in finding global optimisation solutions.

This chapter showed both GA and PSO are quite capable of finding feasible and near optimum solutions without much difficulty, but both of these methods showed unequable suitability in different circumstances. There were some findings in benchmarking these two approaches in both mathematical and engineering applications. First, GA tends to improve its performance quicker by increasing the number of populations in a generation, while PSO prefers to enlarge the number of iteration runs for the sake of promoting the quality of a global optimisation solution. Second, within the same number of computation loads, PSO showed better and more reliable performances in engineering applications than GA. Third, it is also demonstrated that the time requirement for these two approaches under the same computation is different. Finally, this chapter concluded that the PSO would be more suitable specifically in the application of offshore optimisation.

6 EVOLUTIONARY OPTIMISATION STUDY OF OFFSHORE MOORING DESIGN

6.1 Introduction

The design of mooring cable structures is considered to be a difficult and laborious task, which in most of cases produces sub-optimal results. Classical search approaches have limited the scope of the applications since they tend to converge to local optimum [24]. Evolutionary design methods have the potential to solve difficult design problems in terms of finding global optima and efficiency [38], and even though some open search evolutionary approaches have a comprehensive ability to find optimum solutions, the complexity of designing mooring cable systems presents further challenges to find feasible, if not optimum, solutions in many circumstances. The trend for practical offshore cable designs to meet strict and harsh constraints is increasing; in many EA optimisations they start with a wide open searching space and then narrow it down to a feasible and optimised zone. However, with the increasing number of constraints, searching for feasible solutions becomes more and more challenging.

This chapter describes a new method of design and optimisation using one of the stochastic search methods – particle swarm optimisation (PSO). A PSO has been selected after a thorough study of the proposed mooring design and the abilities of various open search algorithms. A comparison of PSO and GA is presented in Chapter 5. The new method described in this chapter incorporates a cost-economical optimisation (to minimise material) and a performance based optimisation. A prototype evolutionary mooring design software that uses multi-objective particle swarm optimisation (MOPSO) for a performance based global search, has been developed. The components, philosophy, and operation of the software are discussed in detail in this chapter. This method can make multiple feasible solutions available for different performance where the design engineer remains in charge of the selection.

A “real life” mooring case study has been presented to examine and compare the PSO and MOPSO. The process of improving searching under real practical

constraints explores the complexity of offshore mooring designs. The limitation of the application of PSO, and the strength of the application of MOPSO in the offshore field have been revealed through searching.

6.2 Optimisation Software and Operation

The optimisation mooring design software is written in the Matlab language. The original PSO code was developed by author from theoretical foundation. Meanwhile, the code was verified and benchmarked via real engineering applications. Details are referred to Chapter 4. The reason for developing original code is to fulfil the function of customisation, full understanding and control. The MOPSO code was modified with Matlab for consistent reasons.

The high level of the algorithm for the software has been summarised as a flowchart, which is presented in Figure 6.1. The results of the simulation were designed as an external block in the software. Under this special structure, it is easy to communicate between the optimisation algorithm and other software packages, apart from OrcaFlex. If another simulation package is preferred, such as Ariane 7, the external simulation block can be simply replaced by the preferable ones. Another advantage of this structure is its accessibility and expandability because if necessary, other software packages can be treated as plug-ins.

Figure 6.2 shows the high level structure of the optimisation software with the MOPSO approach. One of the main distinctions with the PSO is an external repository which records all the Pareto optima information (refer to 4.5.2 for details). All the information on the Pareto optima particles is kept in this repository, with their corresponding penalties. The selection of a global best particle is fully customised via the users' preference. Final selection of the solution is made through a full exploration of the external repository. This enables the philosophy of performance based design to be applied, where one parameter can be traded off against another.

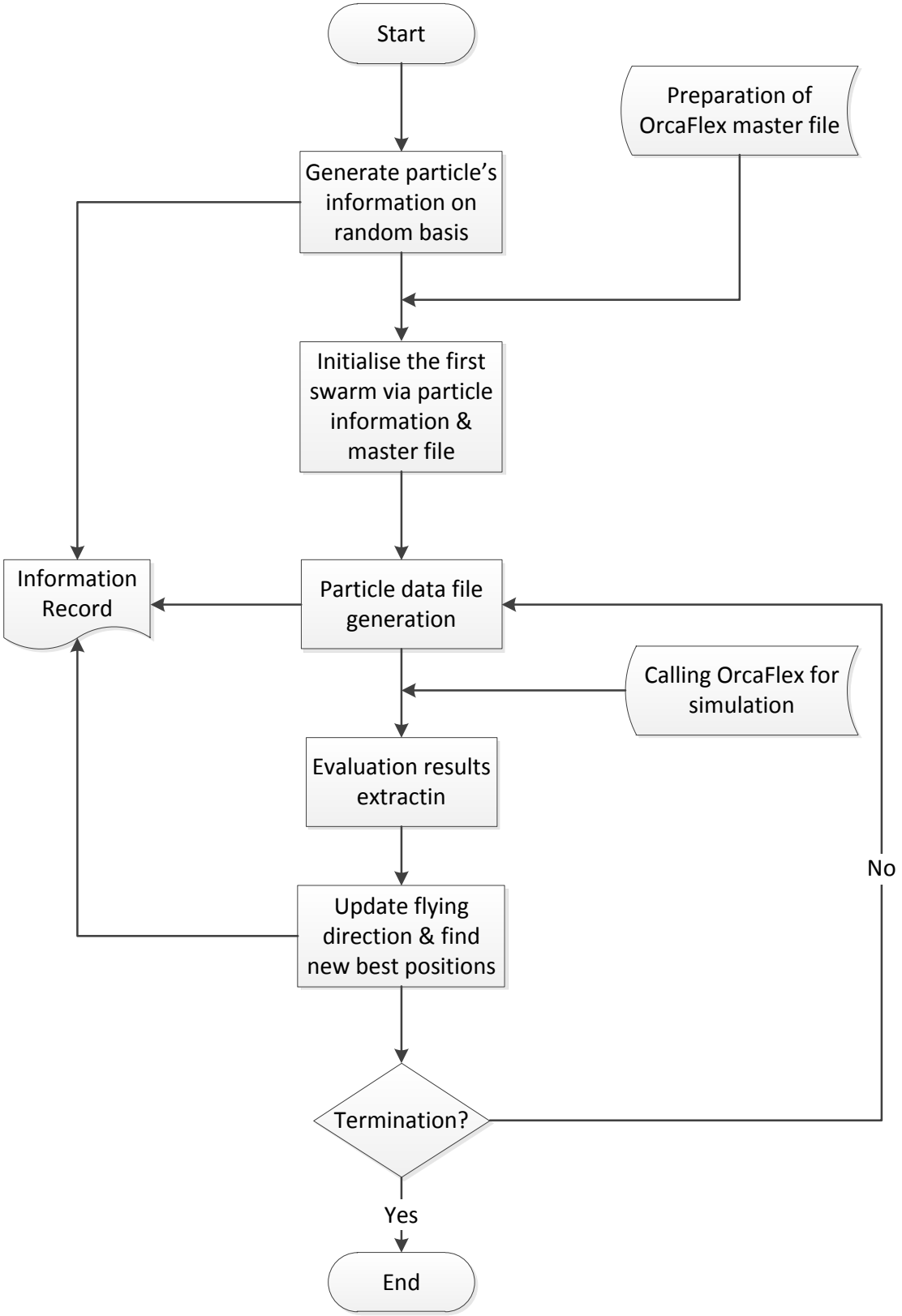

Figure 6.1 High-level flowchart of PSO software

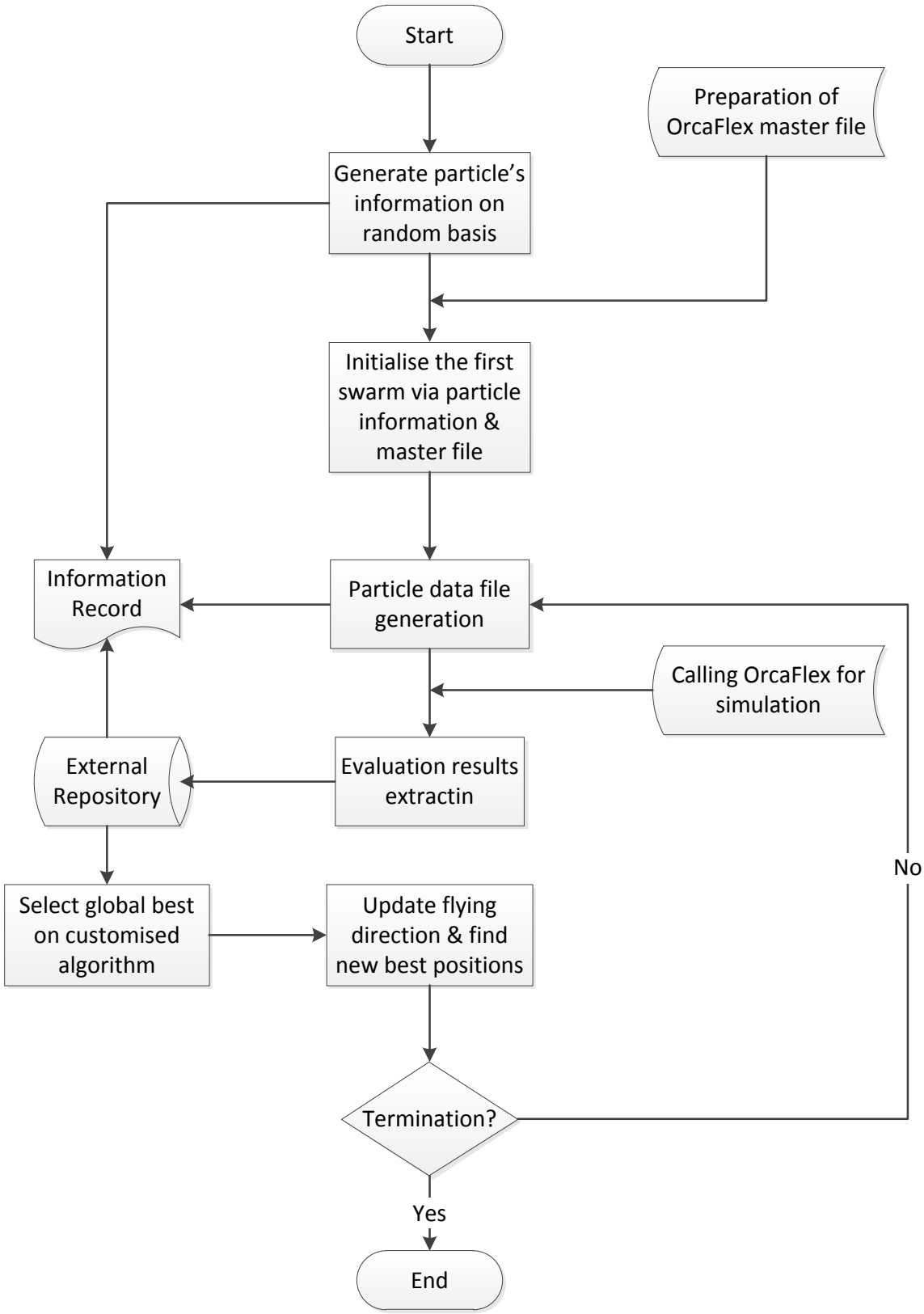

Figure 6.2 High-level flowchart of MOPSO software

6.3 Performance based design & optimisation

6.3.1 Insights into Performance optimisation

In canonical mooring design and optimisation, one particular objective such as economical cost, is focused on by engineers, while other concerns are either dealt with as constraints or design variables. This gives engineers the solutions expected to the main interest but, other factors that can potentially improve the performance of the system have always been neglected. Thus, when the objective for optimisation has been selected, there is always a unique and determine global optima. The sub-optimums that are worse in this particular objective may have strengths and benefits in other aspects that are omitted in this design and optimisation.

The performance based optimisation proposed here allows the system to accommodate more than one objective during the search, while all the relative information from other factors with potential for improvement are retained for a 'trading off' performance between different concerns. For example, the cost of a project may be the first priority in many circumstances, but safety is also important. In canonical mooring design and optimisation, the sub-optimal solutions for the cost that may have a larger factor of safety are ignored by the algorithm itself. However, it is worthwhile exploring these economical sub-optimal solutions for designers, because the potential for improving the safety of a mooring system is worth the investment when a balance between safety and money can be achieved.

In a mooring design and optimisation, there are many mechanical concerns, such as the breaking load of the mooring, anchor tension, payload and, etc., so it is suggested that all the important concerns regarding performance should be investigated during optimisation. In circumstances where one mooring solution has a small cost benefit over another, while the other has a large payload capacity, these two potential solutions should be reserved for final selection by the clients. This design philosophy gives clients more choice when balancing different aspects of the system. This type of design is illustrated in section 7.4.

6.3.2 Limitation of penalty functions

In some circumstances, engineers struggle to find feasible solutions to many offshore applications. This is due to the number and characteristics of practical constraints in a project. A large number of constraints and conflicting constraints may lead a problem being challenged in an open search optimisation field because of the penalty schemes that are applied.

Evolutionary optimisation approaches are open search algorithms that are assisted by constraints handling schemes. As reviewed in section 4.5, the scheme most commonly seen is the penalty. A penalty scheme is triggered when there is a constraint violation, and where the extent of the penalty is associated with the severity of the violation. Generally, the more a constraint is violated, the heavier the penalty that is applied. There is more potential for the constraints to be competing with the increasing number of constraints.

A penalty scheme is useful and sufficient when the number of constraints is small to medium, although the constraints should not be conflicting or competing. The application of a penalty scheme in open search optimisation involves tedious trial and error activities, and hence, within the approach to evolutionary designs, finding the appropriate penalty scheme for an engineering optimisation is not an easy task. The first difficulty is that most offshore engineering problems have a complex response surface in nature. Apart from its complexity, the penalty added onto the objective function increases the unpredictability of the response surface. When engineers do not fully understand the constraints in a new project, the penalty scheme may increase the uncertainty of the total response surface.

One of the limitations of applying the penalty scheme is that it cannot distinguish between superior or inferior solutions with the equivalent fitness values that were penalised, and regardless of the number of components the penalty has, as long as they have equivalent total fitness values, the scheme treats all the candidates equally. However, in practice this is not the truth.

Another limitation of the penalty scheme is that it cannot distinguish between solutions with the same penalised values. For example, if the scheme penalised a candidate due to a violation of constraints A and B to the value of 20 and 30 respectively, that would be equivalent to penalising them for the value of 15 and 35, since the total penalty values are both equal to 50 for the two cases. Therefore, if a complex problem involves numerous constraints, a single penalty scheme cannot guarantee to find the feasible and/or optimum solutions.

The highly constrained problems using the penalty scheme generally require tedious calibration and tuning based on trial and error. Finely tuned penalty functions can lead optimisation to the direction of convergence. While poorly created penalty functions may lead the search astray and throw unexpected solutions without any warning. However, in the field of offshore engineering, many problems are struggling to comply.

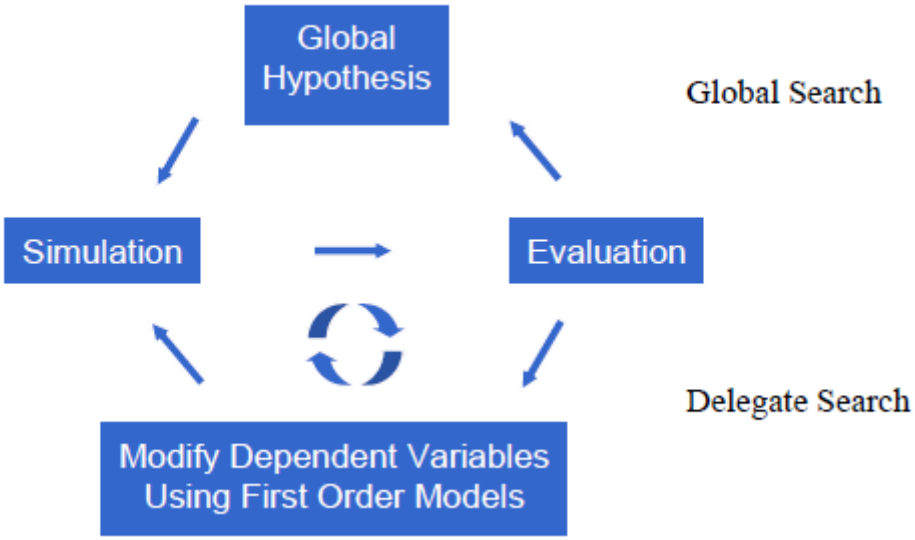

Figure 6.3 Delegate search using the two part optimisation process [139]

Cunliffe [139] claimed that using the penalty system is not an exact science in SCR optimisation. An approach that breaks the problem into independent and dependent variables to form a second, inner, delegate search algorithm was introduced. The technique ran both dependent and independent variables search in parallel and hybrid (as shown in Figure 6.3). This method requires a high level understanding of the problem of optimisation for the sake of categorising the dependent and independent

variables. The acquisition of dependent and independent variables requires pre-analysis of the problem before implanting optimisation.

Le Huédé et al [113] introduced a multi-constrained search (MCS) for multi-criteria combinatorial optimisation problems. This algorithm interchanges a few branch & bound searches with diverse search strategies. The MCS relies on one search strategy per criterion. The main idea of the MCS is to alternate searches on the problem criteria to improve the global fitness every time a solution is found. Essentially, it is a repeated mono-criterion search with a feedback to guide through the search.

However, the penalty scheme used in optimisation algorithms is generally considered to be appropriate with the limited number of penalised items. In order to keep using an evolutionary optimisation algorithm in offshore engineering in a more robust manner, it should have, but not be limited to the following features:

- A searching algorithm can be carried out without a detailed acknowledgement of the problem
- Searching algorithm should be able to search in an open space
- It can handle multi-criteria
- Penalty scheme should not have too many components (no more than three)
- Both violated and complied constraints are recorded
- It can provide multiples of feasible candidates.

6.3.3 Constraint management

The design and optimisation of mooring systems requires the management of constraints from two sources. The first one is the constraints from the optimisation algorithms itself, and the other is from the mooring problems.

6.3.3.1 Constraints from mooring systems and their management

Most evolutionary optimisation algorithms are defined for unconstrained problems. There are a number of ways for handling constraints, including repairing, rejecting, modifying, and penalising[88]. These strategies have been discussed in section 4.5.3. The most common strategy used to accommodate the inclusion of a violation is the penalty scheme. Therefore, when applying the mooring optimisation, the penalising

strategy has been adopted in this research, but with a limited number of penalised items. This technique translates any of the defined violations to a measurable quantitative addition to the value of the objective function, which enables the algorithm to discern which particle is better. The methods for constructing the penalty function are various, but they are generally tweaked on the basis of trial and error to fit the purpose of the problem.

The design constraints of the optimisation problem are found from an analysis of the mooring configurations. The structural and applied constraints are as follows:

- The maximum tension acting on the mooring cables (to assure the integrity of the mooring system such that the mooring legs do not break)
- The maximum excursion of the vessel at a critical environment (governed by the restoring force that the mooring system can provide)
- The minimum requirement of the vessel's payload (can be expressed as an index by the vertical components of the mooring tension
- The maximum tension from the anchors (to avoid anchor failure)

A penalty function associated with a violation of the constraints is defined as a ratio of x between the simulated values and the constraint limits, as follows:

$$P = \begin{cases} \sum k \times (x), if\ x \geq 1 \\ 0, \quad if\ x < 1 \end{cases} \tag{6.1}$$

where k is the coefficient associated with each criterion. The factor k is the penalty threshold that is the initial penalty when any constraints have been violated. It was applied as soon as the penalty scheme had been triggered. It is a quantitative criterion that distinguishes the boundary between feasible and constraints-violated particles. The value of the k factor is associated with the tolerance of emerging non-constrained solutions.

The penalty function associated with the shortage of a minimum capacity requirement is defined as a ratio of y between the simulated values and the minimum requirement threshold values, as follows:

$$P = \begin{cases} \sum k \times (y), if\ y < 1 \\ 0, \quad if\ y \geq 1 \end{cases} \tag{6.2}$$

6.3.3.2 Constraints from the optimisation algorithm

The positions of all particles have been constrained in a certain range of space according to a particular problem. The range in PSO is referred to as bounds. If particles move beyond the bounds, it can be dealt with by various techniques, as discussed in section 4.5.3, such as limiting the speed of the particle, and relocating or re-selecting the particles. In all, the PSO algorithm does not allow any particles to move outside the bounds.

6.3.4 Global and local best particles selection algorithms for mooring systems

For a PSO algorithm, choosing the best particle is straightforward. The particle with the best fitness value in the swarm is selected as the global best particle. The best fitness value of a single particle over time history is selected as the local best particle. This is the definition of the two items in the PSO.

The selection of the global best solution in MOPSO requires the assistance of Pareto optimal. The details of this technique are referred to in section 4.4. The entire Pareto optima particles are placed in a grid for a roulette wheel type of selection. However, in the field of offshore optimisation, not all Pareto optimal are equally important. Therefore, the global and local best selection methods for the optimisation of offshore mooring system need to be specified.

6.3.4.1 MOPSO global best selection algorithm for mooring systems

Instead of using a roulette wheel selection algorithm, the distances for every Pareto optimal are calculated. In Figure 6.4, the distances from the origin to Pareto optimal are shown. The “origin” point should be manually selected in terms of the problem itself, after careful examination. In Figure 6.4 the “origin” is shown at (0, 0) but the actual location depends on the problem being studied.

Objectives can generally be categorised into two groups. The first group is objectives that do not have a target value, such as the economic cost, which needs to be minimised as much as possible. The second group is objectives that have certain specific expectations. For example, the tension between the anchor and the mooring leg has a constraint limit. From the particle point of view, a lower tension than the maximum permitted gives a bigger factor of safety, but this does not mean the lower

the tension, the better the result. The saving of this type of objective would cause other implicit costs, such as high maintenance and the downtime which must be paid, but which cannot always be considered completely. This is where the performance must be balanced. As long as these types of objectives do not violate the constraint limits, they are considered to be acceptable, but since this type of objective has a target expectation, it is advisable to take a slightly smaller value than the target value as the origin.

The MOPSO global best selection algorithm works out the distances between each Pareto optimal and the "origin" for every iteration. The one with the smallest distance has been chosen as the global leader. For instance, as shown in Figure 6.4, particle B will be selected as the global optimum to lead the swarm in the next iteration.

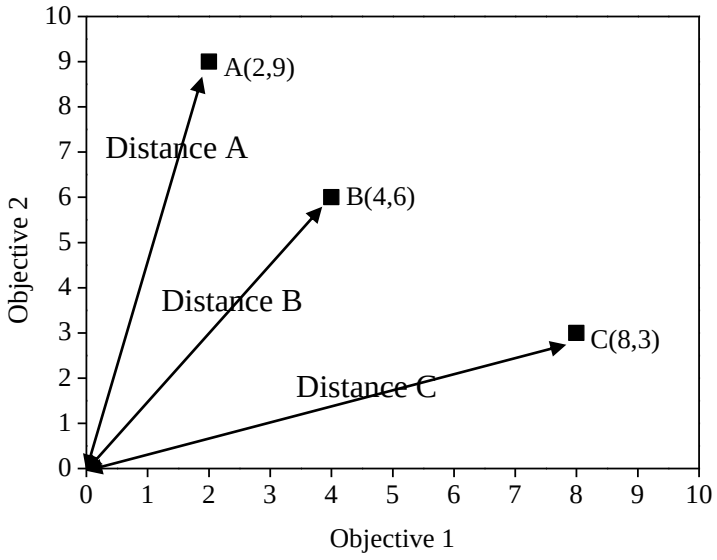

Figure 6.4 Schematic view of global best selection algorithm

Applying this new global best selection algorithm introduces a few benefits. First of all, the origin of the coordinate can easily be adjusted for different requirements and conditions. Second, the axes can be scaled with regards to the level of importance of the different objectives. For example, if the objectives are equally important, the scaling factor can be set as one, but where one objective is more important than

another, a larger scale factor should be applied. Furthermore the coordinates can be extended from one dimension to multiple dimensions to accommodate the number of objectives, and lastly, this algorithm avoided the uncertainties of applying a roulette wheel selection so the MOPSO can perform as the engineers' expected.

6.3.4.2 MOPSO local best selection algorithm

In the Pareto optimum approach, the selection of the local best is based on the Pareto non-dominated positions from its history. If the previous local best position is dominated by its new position, the local best position is then replaced by the new one otherwise the history of the local best position is maintained. If both current and history positions are both Pareto optima positions, the Pareto approach suggests a random selection of any one to be the local best position.

The original Pareto approach is a purely mathematical model based on statistics. In the application of mooring optimisation, the axes are physical representations with dimensions, and therefore the purely mathematical treatment must be modified in the new circumstance. This is the same as the global best selection algorithm where searching for a feasible position is a priority. Therefore, the distances between the Pareto optima to the "origin" can be also calculated. The shorter the distance is, the better the position of a particle. The local best selection algorithm calculates distances for every single particle in a swarm and compares it with the historical best position.

6.4 Optimisation interface and control

OrcaFlex was used for the simulation. There are four types of objects: the generic, environment, line, and vessel, respectively, as described in section 3.4.

6.4.1 Design independent variables

Figure 3.36 presents a typical turret mooring system. The concerns and variables of mooring designs are discussed in section 3.11.2. The independent design variables are normally associated with the geometry of a mooring system.

- Length (L_1), material type and diameter(D_1) of top mooring segment
- Length (L_2), material type and diameter(D_2) of middle mooring segment

- Length (L_3), material type and diameter(D_3) of ground mooring segment

Most mooring legs have three segments, although some have one or two segments, depending on the application. The total length of a mooring leg can be worked out by summation of all the segments. The length of a shackle connecting the segments is negligible. For the type of material, the frequently used material review can be found in section 3.3.1. This parameter is defined in optimisation software as an integer type for selection purposes. The diameters of the segments depend on the type of material selected for that particular segment. For example, the diameters for steel chain can only be provided by the chain manufacturers. In this optimisation, the diameters of the chain are taken from the Ramnäs manufacture manual (Appendix D) where selection is s is based on 33 chain classifications that range from 70mm to 177mm in the production line of Ramnäs.

6.4.2 Design program communication and linking

This section presents details of the process of using Matlab to interact with OrcaFlex via OrcaFxAPI.dll file. The main role of OrcaFlex is to provide data files and simulation result files. All useful information from those files is then extracted into the optimisation program which is coded in Matlab.

OrcaFlex can communicate with other Microsoft Windows based program such as Matlab, via a programming interface. This function is provided as a 32-bit windows dynamic link library (DLL) called OrcFxAPI. It is important to note that in attempting to use Matlab to interact with OrcaFlex DLL, there are both hardware and software pre-requisites. First of all, it should be a Windows based machine because Linux machines or Mac machines are not supported. Second, both the 32-bit version of Matlab (version 2011b at the time of writing) and OrcaFlex (version 9.4f at the time of writing) must be installed on the same computer. Third, The Matlab OrcaFlex interface is accessible. A general illustration of the Matlab-OrcaFlex inter-communication is shown in Figure 6.5.

Initially, the OrcFxAPI.dll creates a Model in Matlab, which is essentially a block of free memory (the grey area in Figure 6.5). Matlab interacts within the Model through a Model Handle, and then an OrcaFlex simulation file is loaded into the free

memory block. Within the simulation file, objects (the green area in Figure 6.5) interact through their own Object Handles. With each object, information is directly linked to their individual variables, represented by the orange area in Figure 6.5. Every variable has a Variable ID so that Matlab can extract the data in the variables. Once all the information in a Model has been extracted, it should be removed from the memory. If new data are required, a new Model is then created and the file loaded into this new memory block. In general, this interactive process can be logically summarised in the sequence of:

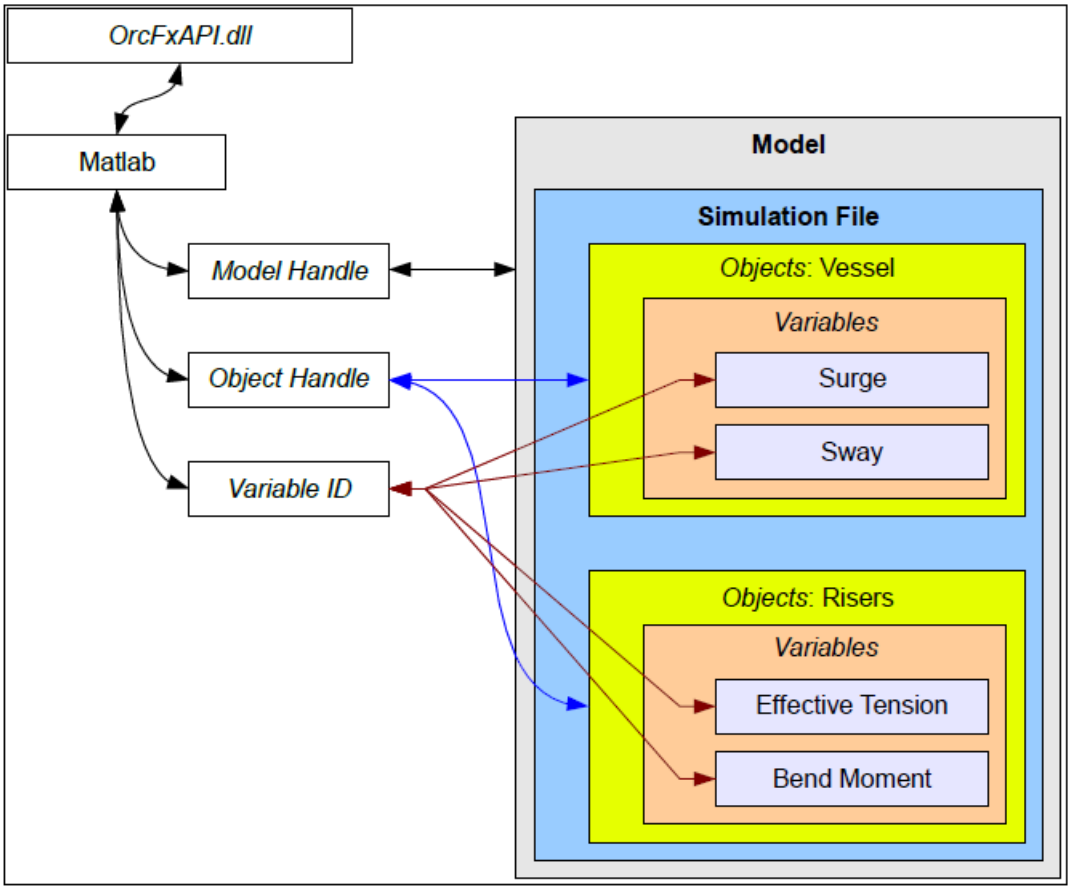

Figure 6.5 Matlab-OrcaFlex inter-communication flowchart [140]

1. Loading OrcFxAPI.dll
2. Creating a Model (create a memory block in a computer)
3. Loading a file into the Model
4. Extracting Object Handles and Variable ID
5. Extracting results that interests users

The OrcaFlex dynamic library link was developed in C language. The use of the library link requires a certain level of C code skills. Data extraction between arguments and returns are through pointers. An introduction of communication between Matlab and OrcaFlex can be found in Appendix A.

6.4.3 Template Master-file

A master file is an OrcaFlex data file designed by operators for the purpose of pre-setting all the non-variable information. A typical master file should be comprehensive enough to include at least one vessel and the corresponding environmental information.

A master file is used to reproduce all other date files to be simulated in OrcaFlex, and therefore its accuracy must be fully checked. All the default information will be duplicated to other data files for analysis. In the case study, the reference case file can be fully functional as a master file. The pre-set non-variable parameters in the master file include:

- Water depth
- Environmental pre-settings
- Vessel information includes its position and other useful information
- Positions of Fairleads (these may be variable in different cases)
- Positions of Anchors (these may be variable in different cases)
- All other general information such as static converge parameters and units, et cetera in OrcaFlex

6.5 PSO and MOPSO parameters and settings

Based on the extensive parametric studies of PSO from de Pina [107], the performance of PSO is affected considerably by using the coefficient of inertia weight. Any value less than 0.8 for the inertia weight results in a premature convergence. For inertia weights larger than w = 0.8, steady mean value and low standard deviations are expected. The author confirmed that the results of PSO were not influenced very much by the cognitive parameter c_1 and social parameter c_2 in SCR design. Perez [93] provided the stability characteristics of PSO through

eigenvalue analysis. Details of the results can be found in section 4.1. Therefore, the inertia weight w is taken as 0.875 in this optimisation software. The cognitive parameter c_1 and social parameter c_2 are taken as 2. The software terminates the execution of PSO at the 100^{th} iteration. The results and comparison of PSO and MOPSO are made in terms of effectiveness and efficiency at the end of the 100^{th} iteration. According to de Pina [107], the PSO considered a population of 10 particles in offshore optimisation. A small particle number would result in insufficent space for exploration and show premature results, although a larger particle number would result in an increase of computational times without significantly improving the final results.

To sum up, the coefficients of PSO used in the software are presented as follows, so that the quality of optimisation could be maximised. The configurable parameters are given below, with the default values in Table 6.1.

Table 6.1 PSO and MOPSO experiment coefficients

Particle number in a swarm	10
Number of iterations	100
Cognitive parameter (c_1)	2
Social parameter (c_2)	2
Inertia weight (w)	0.875

6.6 An overview of a turret mooring case

The studies on the PSO based optimisation in this work were performed for a turret mooring configuration installed at a sea depth of 65 m, with a large FPSO. The purpose of applying PSO based optimisation is to demonstrate the ability of the new method to handle complex real world offshore projects and the characteristics of PSO and MOPSO.

6.6.1 Description of the case study

The case study undertaken here was described in section 3.11.1. The API RP-2SK [13] was used for the optimisation design. The optimisation design software was developed to implement the API stationkeeping standards.

6.6.2 Reference case description and assumption

This section describes the details of the turret mooring system. The solution and results illustrated here served as a reference case for comparison use in the following study and optimisation sections. The offset of the FPSO was limited to 25.5m to accommodate for riser configuration under a fully loaded situation. This offset is assumed to be the result of the excursion from a combination of the dynamic composition of wind, wave, and current in their collinear circumstance. In other words, when the vessel has 25.5m offset, the restoring force provided by the mooring system is in equilibrium with the combined environmental force.

Within the maximum offset of 25.5m, simulation results of all the mooring legs have been summarised in Table 6.2. The other end force column, in Table 6.2, is taken in this case as tensions in the anchors, while friction between the chain and the sea bed are omitted. The maximum simulated tensions were used to calculate the forces necessary to break the cables.

Table 6.2 Results extracted from OrcaFlex for the reference case

Connection to	**Vessel End**		**Other End**	**Max Tension**
	Total force (kN)	**Vertical force (kN)**	**Total force (kN)**	**Tension (kN)**
Line1 Top End	440.68	364.35	392.35	440.61
Line2 Top End	396.63	338.97	385.98	396.57
Line3 Top End	362.73	318.72	381.44	381.31
Line4 Top End	222.43	219.94	365.16	365.03
Line5 Top End	226.86	223.74	365.54	365.41
Line6 Top End	230.58	226.74	365.94	365.81
Line7 Top End	1884.44	917.20	1475.76	1884.41
Line8 Top End	2094.12	967.96	1709.57	2094.09
Line9 Top End	2305.01	1015.59	2055.59	2304.98

Figure 6.6 and Figure 6.7 show snapshots of the mooring system in OrcaFlex under equilibrium, in both the front and top view angles respectively. Figure 6.6 indicates the total 25.5m offset to the direction of x in positive direction, with the turret mooring in equilibrium. The white points in the graph are the points the mooring chain touches the seabed. The mooring induced reduction in payload on the FPSO can be represented by the vertical components of the mooring system force. The

smaller the vertical forces are the larger payload the vessel can have. Therefore, the vertical component of the tension from the mooring system can be considered as payload reduction index of the vessel.

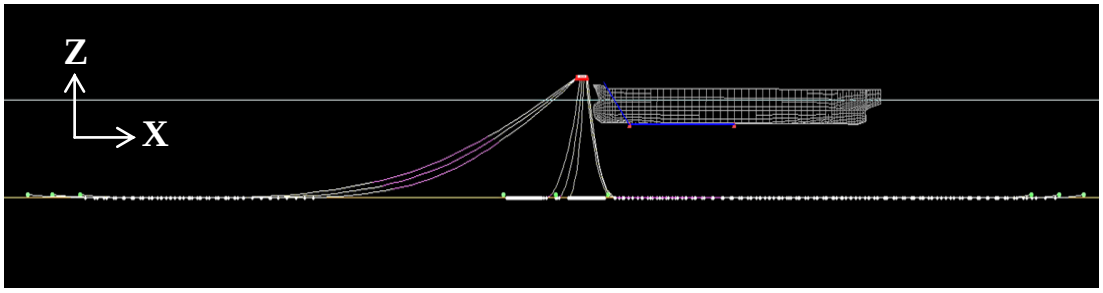

Figure 6.6 Front view of the reference mooring system under equilibrium

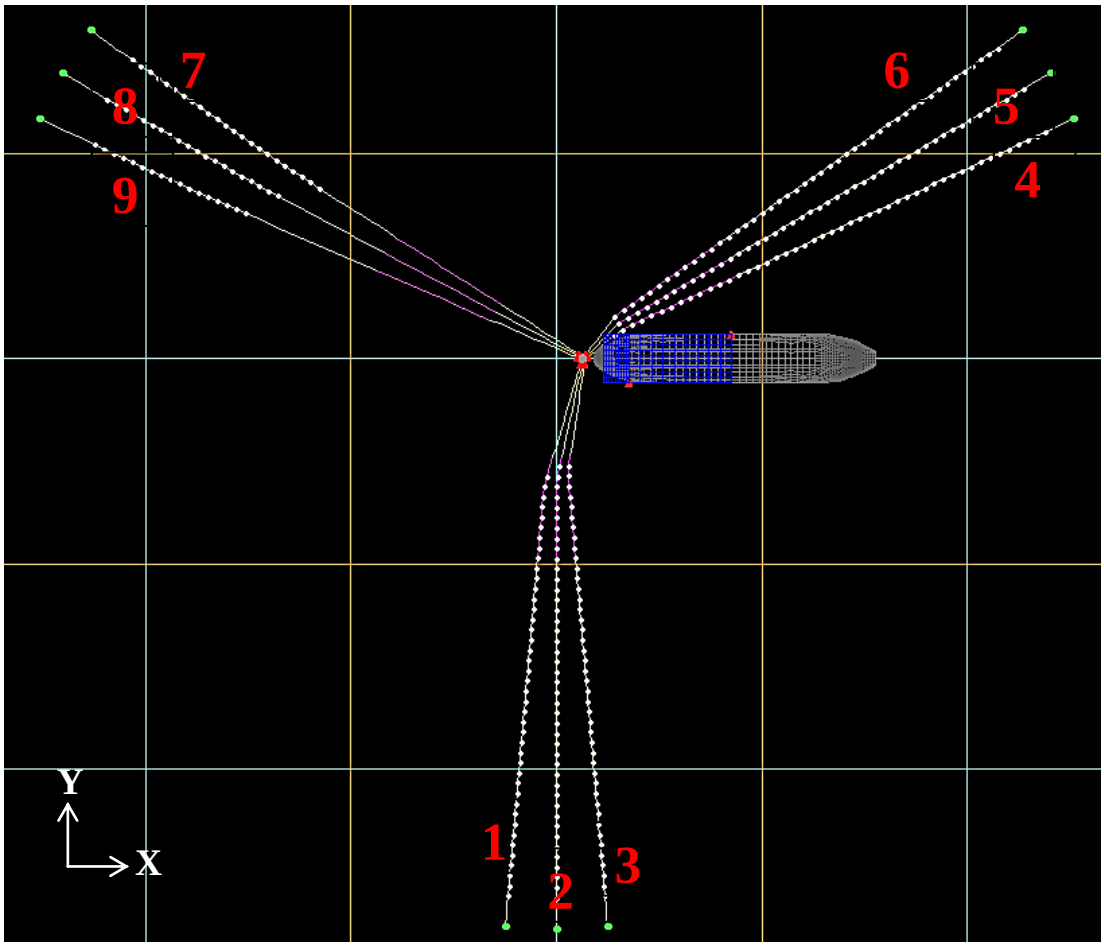

Figure 6.7 Top view of the reference mooring system under equilibrium

The total external environmental force, payload index, and maximum anchor tension have been summarised in Table 6.3. The mooring leg breaking force varies on the

diameters of the moorings. The larger the diameter of the bar of the mooring chain, the greater the capacity of the mooring leg.

The total depth of water of the system is only 65m, which indicates a shallow water environment. In such depths, section 3.9 suggests that a chain type mooring be used. Therefore, the mooring system is optimised based on the same grade of chain material for simplicity.

Table 6.3 Mooring system capacity

Force (kN)		
Environmental force	**Payload index**	**Max anchor tension force**
4990.32	4593.21	2055.59

As discussed in section 3.11.2, the mooring case overview has highlighted a few concerns, which can be monitored in the process of optimisation.

- The material cost of this mooring system can be simply represented in terms of the volume of steel. The current total volume of steel for the mooring system is 187.6 m3 (calculated from a combination of the lengths and diameters of the mooring legs in Table 3.5)
- The maximum amount of tension acting on mooring cables, between all nine legs, appears in leg number nine, which is 2304.98 kN. The maximum tensions from all the mooring legs are monitored.
- The total excursion of this mooring system is 25.5 m to the *x* positive direction. Within such an excursion, the total peak environmental force provided by the mooring system is 4990.32 kN, by assumption, which is also assumed to be equal to the horizontal restore force provided by the mooring system.
- The maximum requirement of vessel payload reduction is represented in terms of payload index of the mooring system at 4593.21 kN.
- The maximum anchor tension format, as the other end tension from mooring leg number 9, is 2055.59 kN.

The design variables described in section 6.4.1 related to this mooring case are given limitations in Table 6.4. The upper and lower limit length boundaries are set to avoid excessive lengths in mooring legs that lead to simulation failed (These would trigger

the death penalty scheme in the optimisation). The diameters of the chain can be selected arbitrarily from the Ramnäs product manual. This selection of chain diameters is based on 33 indices of the chain classifications that range from 70mm to 177mm in the product manual of Ramnäs.

Table 6.4 Limits range of segment lengths

	Minimum	**Maximum**
Top segment length (L_1)	114 m	130 m
Middle segment length (L_2)	88 m	105 m
Ground segment length (L_3)	348 m	365 m
Top segment diameter (D_1)	70 mm (Index 1)	177 mm (Index 33)
Middle segment diameter (D_2)	70 mm (Index 1)	177 mm (Index 33)
Ground segment diameter(D_3)	70 mm (Index 1)	177 mm (Index 33)

6.7 Chapter summary

This chapter has presented the offshore cable structure optimisation software which has been written as an essential component of this project. The software uses Matlab to communicate with the industry package OrcaFlex, to control, simulate, and extract. For the purpose of simplicity, except for the master file preparation, all other control variables and parameters took place within Matlab, OrcaFlex was only used to provide results. Hence, the evolutionary software itself is flexible enough for block modification and communication with other commercial packages.

This evolutionary software is determined from a comprehensive review of the mooring design search and a detailed review and comparison of the strengths and limitations of a wide range of search techniques. The optimisation software on the basis of MOPSO has been modified without sacrificing any merits of PSO. A fully customised external expository helped the performance based design to a feasible level. The limitations and drawbacks of PSO optimisation in the heavily constrained offshore problems have been overcome, so only small acknowledgements are requirement before taking the problem to optimisation or feasibility searching.

7 RESULTS & COMPARISON OF THE EVOLUTIONARY OPTIMISATION STUDY OF THE OFFSHORE MOORING DESIGN

7.1 Introduction

This chapter introduces the use of penalty functions in PSO and MOPSO. Once the penalty levels have been established, the optimisation results from the PSO and MOPSO algorithms are compared and contrasted for the mooring case study described in Chapter 6. The penalty functions are shown to be critical in the success of PSO. The MOPSO algorithm is shown to be more robust and able to produce a good solutions with a range of performance characteristics.

In parallel with the application of MOPSO, the concept of performance based MOPSO has been explained and illustrated with examples of how engineers can apply this approach in offshore mooring design.

7.2 Description of the experiments

The mooring case presented from section 6.6.1 and 6.6.2 is studied with the PSO and the performance based MOPSO method. The studies focused on the convergence of the optimisation algorithm for the offshore mooring application.

In the PSO, the convergence is related to the penalty functions. In order to obtain a healthy behaviour, three different penalty functions were used to compare their convergence. The details are covered in section 7.3.2. To compare the results with those obtained by the performance based MOPSO method, the convergence and efficiency were assessed with the same number of iterative runs presented in section 6.5. In order to avoid the uncertainty due to the randomly generated initial populations, the optimisation algorithm for both PSO and performance based MOPSO were conducted six times for each set of penalty functions within the same coefficients.

7.3 Discussion of PSO studies and results

7.3.1 PSO fitness or objective function

It is common in engineering optimisation problems that the objective function is associated with the lowest cost. The compliance of other conditions is defined in the penalty functions.

For the mooring problem defined in section 6.6, the cost function can simply be taken as the cost of materials. Therefore, the cost function of the mooring problem is given by:

$$f = \sum_{i=1}^{9}\left(\frac{\pi D_{e,i}^2}{4}\times L_i + P_{B,i} + P_{R,i} + P_{P,i} + P_{D,i}\right) \tag{7.1}$$

where D_e is the equivalent diameter of the cable/chain, L is the length of the cable, P_B is the value of penalty function due to cable breaking load violations, P_R is the value of the penalty function corresponding to the restoring forces violations by the mooring system, P_P is the value of the penalty function as to payload violations, and P_D is the penalty value as per anchor tension violations.

The fitness function (sometimes called objective function) defines the quality measurement of a particle in a swarm. From an optimisation point of view, the particle with a smaller fitness value represents a superior quality. The fitness value is the only criterion to guide through the search of optimisation. The cost function of each particle is calculated based on Equation (7.1). Finally, for this mooring problem, the mathematical function of the fitness function is as follows:

$$fitness = \frac{f}{f_{ref}} \tag{7.2}$$

where f_{ref} is the value of the cost function from the mooring system introduced in section6.6.2, where it is a constant over the optimisation. The purpose of dividing the cost function by the reference from the mooring system is to normalise the expression. Thus, the fitness is scaled down to a dimensionless value to the reference.

In terms of optimisation a fitness of less than 1, and without any penalty, indicates that the new system is superior to the reference case illustrated in section 6.6.2. If the fitness is less than 1 but it has penalty values, this indicates that one or more constraints have been violated. In this case, the optimisation either has an inappropriate fitness or inappropriate penalty scheme. It is not acceptable to have any penalty in the final solution. In circumstance where the fitness is larger than 1 and it has no penalty values, this depicts a feasible system but one that is not as good as the reference system. For this situation, the optimisation scheme is deemed to have failed to fulfil its purpose. In summary, with a fitness value that is less than 1 and a penalty value equal to 0, is the goal of the optimisation.

7.3.2 PSO constraints and penalty functions

The PSO can only have a single objective function values, hence all other concerns listed in section 6.6.2 should be dealt with through penalty functions. From the overview of the penalty scheme introduced in section 6.3.2, finding an appropriate penalty function involves a number of trial and errors. In all, the items that have been penalised includes overloaded tensions of mooring legs, an inadequate restoring force to withstand the environmental force, and insufficient vessel payload and excessive anchor tensions.

The application of the penalty threshold is considered to be necessary in the penalty scheme as it draws a border between feasible and constraint violated candidates. The constraint limits are set according to the reference system. Therefore, the term feasible has been used to describe a system that is being optimised in terms of a mooring reference system. In contrast, the term “constraint violated candidate” has been vaguely defined as mooring systems where any items listed in section 6.6.2 are being violated. However, the term “constraint violated” does not mean the solution is not working, although it may have a worse solution than the reference system.

This factor k in the penalty scheme draws a border between the constraints comply particles and constraint violated particles. Increasing the k value of the penalty threshold, the border of feasible and constraint violated particles becomes more and more distinguishable. Since the healthiness of the PSO is related to the quality of the penalty function, aligned with section 6.3.2, the penalty scheme is designed in three

groups as large, medium, and small penalty thresholds, respectively. This categorisation in terms of the penalty threshold is based on the fitness function of the mooring problem. The small penalty threshold has a *k* factor in the Equation (6.1) and Equation (6.2) of about 10 per cent of the cost of the reference material. While the medium has a *k* factor of about 30 per cent, and the large has an average *k* factor of about 80 percent. A summary of *k* factors is given in Table 7.1 as:

Table 7.1 *k* factors table of mooring penalty functions

	***k* factors of constraint types**			
	Maximum tension	**Environmental force**	**Minimum payload**	**Anchor tension**
Large	100	200	100	200
Medium	50	50	50	50
Small	20	20	20	20

7.3.3 PSO results and discussion

The optimisation results of the PSO presented in Table 7.2 were extracted from the best of six runs. These values correspond to the best solution in three different penalty function groups. It is obvious that a good result of the optimisation algorithm has a low fitness (less than 1) value and no constraints violation. LTP, MTP, and STP stand for Large Threshold Penalty function (Group I), Medium Threshold Penalty function (Group II), and Small Threshold Penalty function (Group III) respectively.

From Table 7.2, all the fitness values are less than one however, only Group I (LTP group) has a full compliance of constraints. Thus, the solution from Group I (LTP group) has been optimised. Even though the other two groups have a fitness value of less than one, they have failed to reach optimisation due to violations in the environmental force (offset excursion criterion). In terms of the results of the mooring legs' configurations, Group II and III (MTP and STP group) tend to use the smallest diameter of mooring chain to save materials. The reasons can be investigated by the following plots from Figure 7.1 to Figure 7.3. They examined the convergence, behaviour, and efficiency of the PSO.

Table 7.2 Classical PSO global optimum results extraction

	FV	Mooring leg configurations*	BV	EV	PV	ATV
REF	1	{125,100,360,116,152,108}				
Group I (LTP)	0.616	{129, 98, 354, 90, 147, 73}	N	N	N	N
Group II (MTP)	0.694	{124, 89, 364, 70, 70, 70}	N	***Y***	N	N
Group III (STP)	0.464	{125, 89, 363, 70, 73, 70}	N	***Y***	N	N

Note: REF, the mooring system before optimisation
FV, Fitness Value
BV , mooring leg tension capacity violation
EV, environmental force violation
PV, payload violation
ATV, anchor tension violation
Y, violated
N, non-violated
*, mooring leg configuration in the order of {top segment length in m, middle segment length in m, ground segment length in m, top segment diameter in mm, middle segment diameter in mm, ground segment diameter in mm}

Figure 7.1 shows the normalised objective fitness value of the global best particle information versus iterations in different penalty functions. The NOF stands for Normalised global Objective Fitness values. The fitness values have been normalised to the reference mooring case. Three curves represent Group I (LTP group) in red, Group II (MTP group) in black, and Group III (STP group) in blue, respectively.

It is obvious that for every single run, the normalised objective value of the global best particle decreases as the number of iterations increase. In other words, the optimisation algorithm is finding better and better candidates. The optimisation software has been set to terminate at the 100^{th} iteration since little improvement had been found afterward that. For Group I (the LTP group), two out of six runs have normalised objective fitness (NOF) values less than one. For Group II and III (the MTP and STP groups), all of the runs have an NOF value less than 1. However, it is impossible to draw the conclusion that the solution is optimised to reference without examining their compliance of constraints.

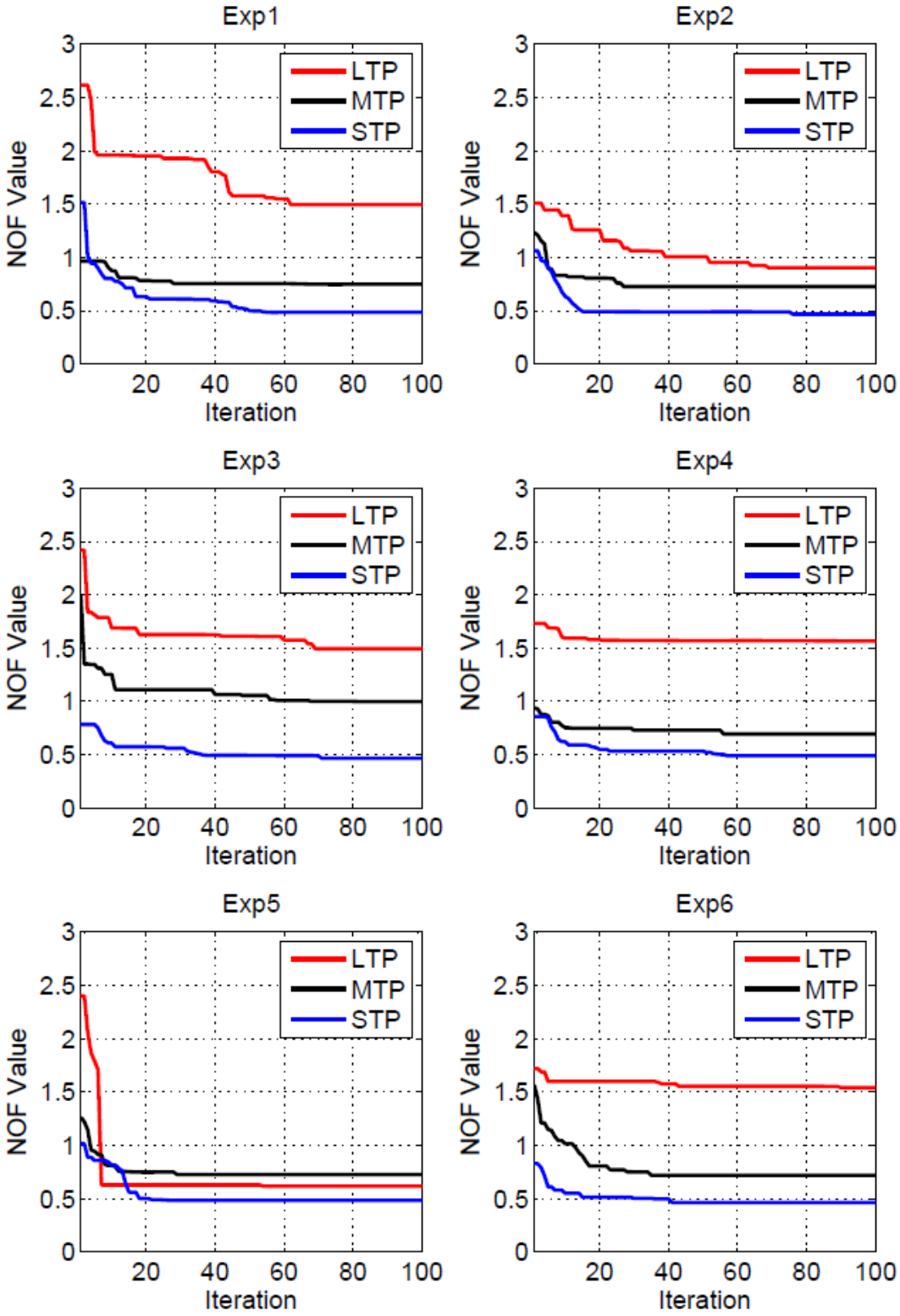

Figure 7.1 Normalised global objective fitness value versus iteration for different penalty function groups

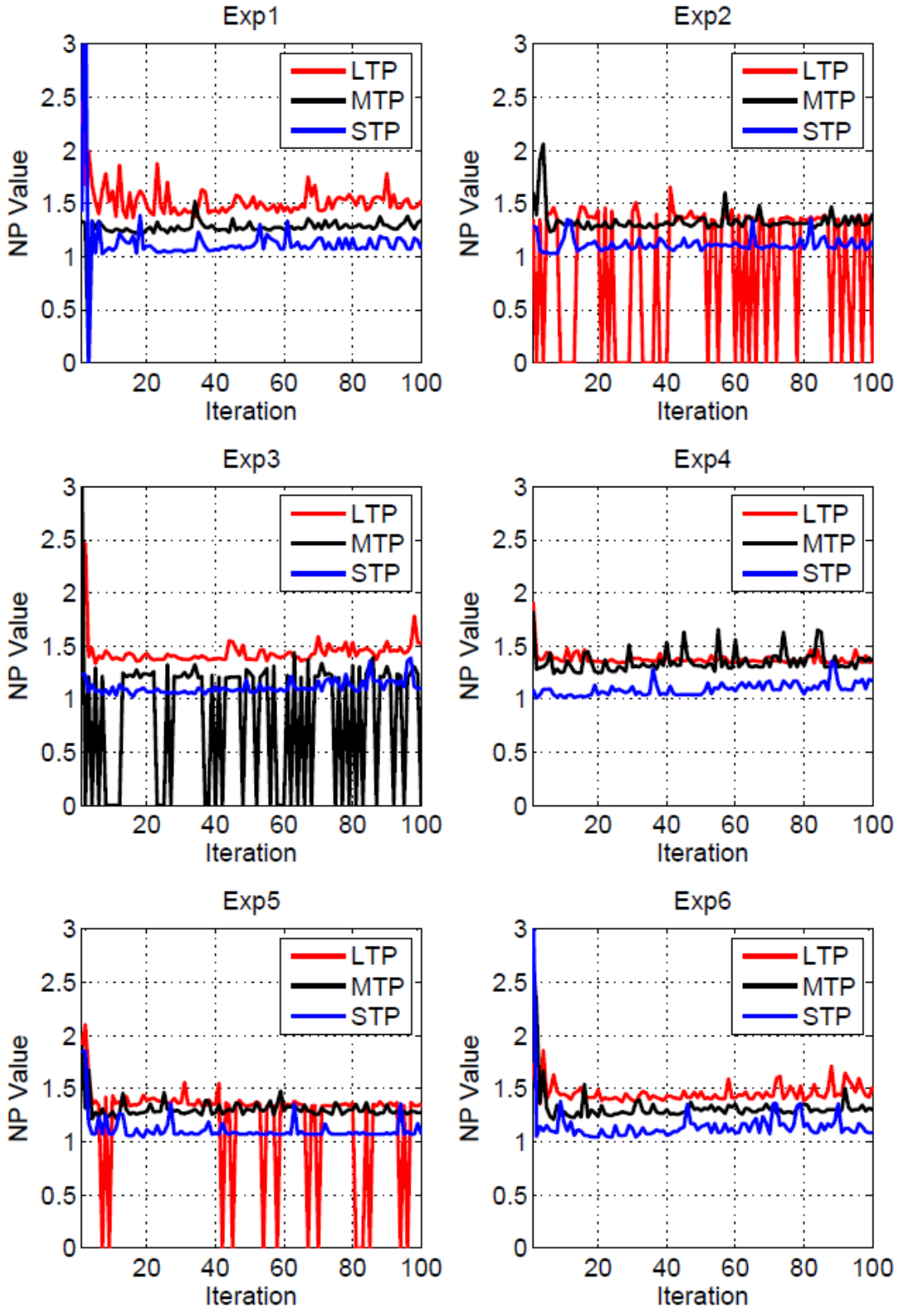

Figure 7.2 Normalised penalty value versus iteration for different penalty function groups

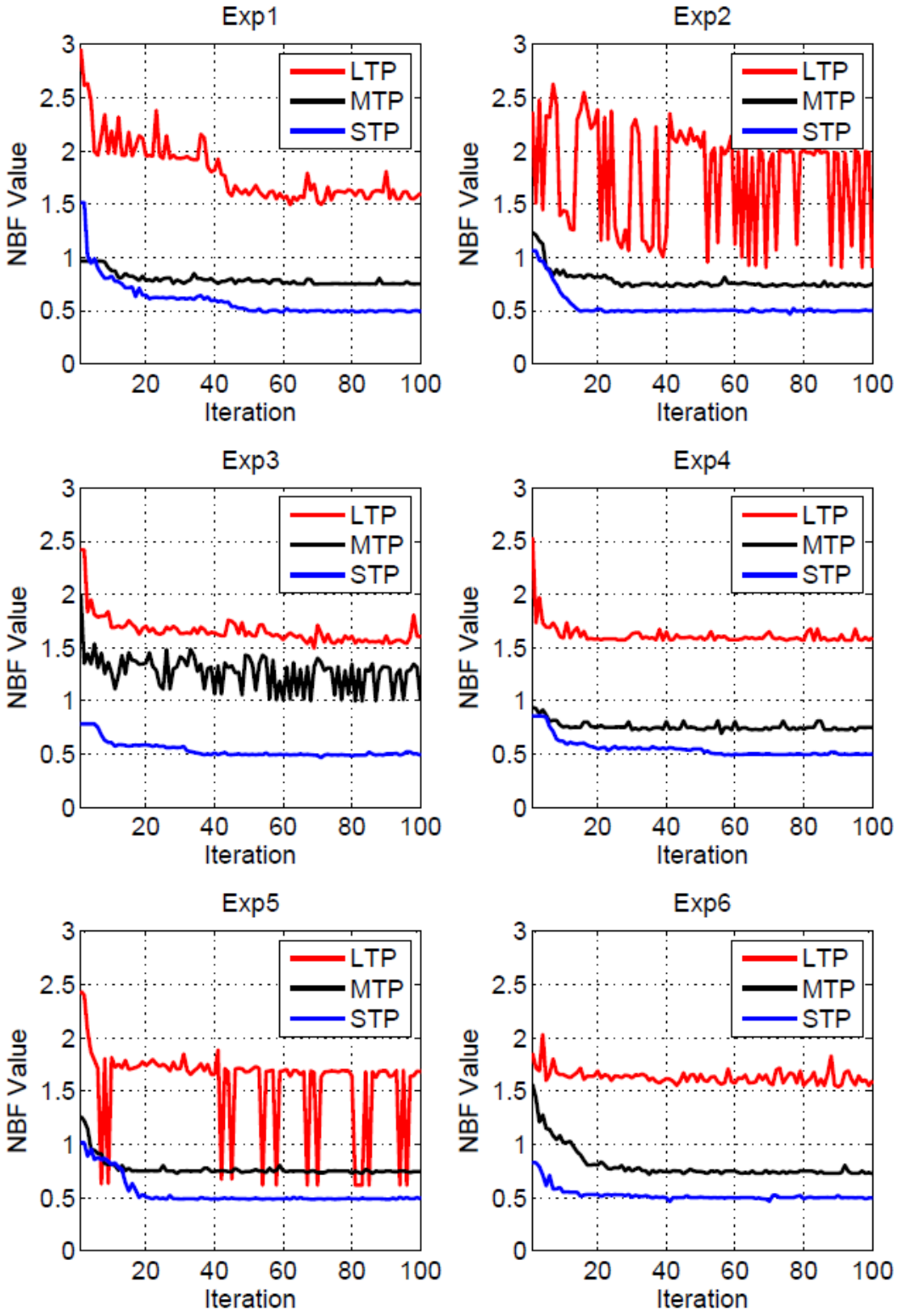

Figure 7.3 Normalised best fitness value versus iteration for different penalty function groups

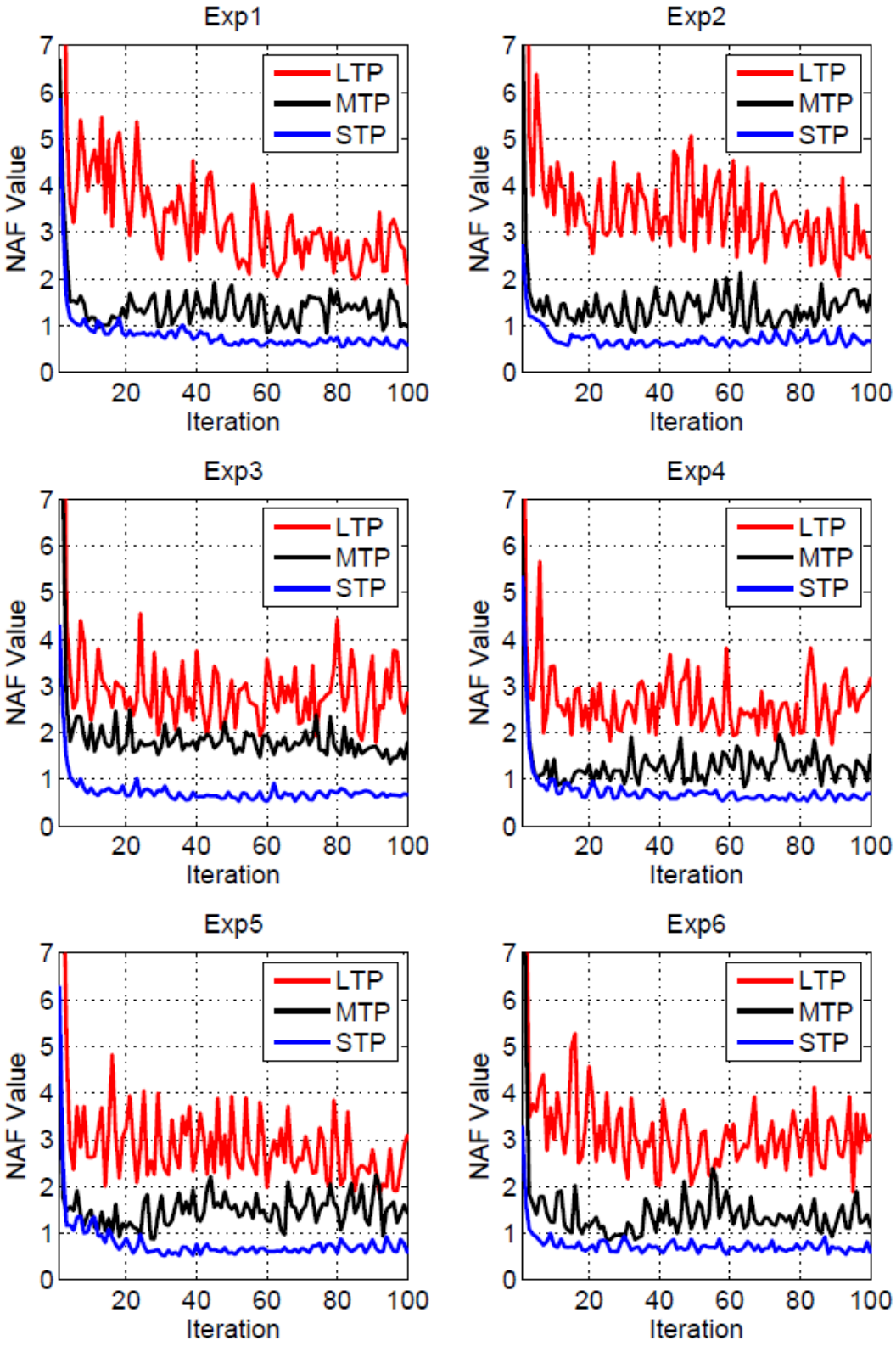

Figure 7.4 Normalised average fitness value versus iteration for different penalty function groups

Another obvious general pattern from Figure 7.1 is the relative locations of the three curves. Except for experiment No. 5, all the other runs have the Group I curve (LTP curve in red) on top, the Group II curve (MTP curve in black) in the middle, and the Group III curve (STP curve in blue) at the bottom. This is due to different levels of the penalty threshold value. The large penalty threshold gives larger penalty values when constraints are violated. Thus, the red (Group I, LTP) curve located on the top of the other two. Nevertheless, this is not a favourite pattern because the normalised objective values are greater than one for most of the runs. In other words, optimised solutions have not being found. In experiment No.5, where the results have been optimised, the red (Group I, LTP) curve crossed over to the black (Group II, MTP) curve.

Figure 7.2 is the graph showing a normalised minimum penalty value versus iteration in all groups of penalty functions. The NP in the diagram represents the Normalised minimum Penalty. The minimum penalty values of Group II and III are normalised to the *k* factors listed in Table 7.1. The Group I (LTP curve) is normalised to an average of the *k* factors in Table 7.1. From this plot, there were three runs out of six (Group I has run No. 2 and 5, Group II has run No. 3) recorded of fully compliance particles. Unfortunately, one was not selected as the best particles due to:

- The value of fitness of mooring optimisation varies in a wide range;
- The penalty threshold is not large enough to clearly identify the border between optimisation and sub-optimal.

To summarise the results, only Group I (LTP group) found an optimised solution. Both Group II (MTP group) and Group III (STP group) presented either sub-optimal or worse solutions to the reference case.

Figure 7.3 plots the normalised best fitness value in a swarm versus iteration for all types of penalty function groups. NBF stands for the Normalised Best Fitness in a swarm. It generally has a similar pattern to Figure 7.2. The best fitness is associated with the minimum weight and penalty values, as particles with a lower penalty can have better fitness within the same weight.

The convergence of the optimisation algorithm can be evaluated based on the average fitness of a swarm versus iteration, as shown in Figure 7.4. It is expected that with the increasing number of iterations, the average fitness value should have a decreasing trend. However, it is clear that both Group II (MTP group) and Group III (STP group) curves have a relatively flat pattern with steady normalised average fitness values. In contrast, Group I (LTP group) has a more fluctuating pattern due to the large threshold penalty function. Hence, the particles violating constraints have a much larger fitness value than Group II (MTP group) and III (STP group). Nevertheless, the Group I curve in this graph seems to have a random pattern instead of a general decreasing trend.

The trigger of penalty is due to a violation of either an inadequate restoring force to withstand the environmental force or excessive anchor tensions, by examination. It is clear that the restoring force provided by the mooring system and anchor tensions is from the two ends of the mooring lines. The top end mooring force provides a restoring force component, and the bottom end force gives the anchor tensions. They are closely linked to each other. Therefore, these two constraints are either conflicting or competing with each other. A more sophisticated approach is warranted to prove this point. The following sections manage the mooring optimisation by MOPSO.

7.4 MOPSO studies and the performance based design

7.4.1 MOPSO fitness or objective functions

MOPSO has the ability to accommodate two or more objectives. To demonstrate its performance in this case study, two objectives were chosen. The first objective is the same as the PSO, which is to minimise the cost of materials, so the fitness function of the first objective is the same as Equation (7.2).

The second objective can be selected from the constraints listed in section 6.6.2. Based on the results discussed in section 7.3.3, the most frequently seen violations were either an inadequate restoring force to withstand the environment or insufficient

anchor tension. For the purpose of proving whether these two criteria are competing, the second objective adopted for optimisation is the anchor tension. As discussed in section 6.3.3, the management techniques are different for different types of constraints.

As described in section 6.3.3, it is unwise to minimise an objective with a certain level of expectation to the greatest extent, because minimising that objective value to the greatest extent would implicitly increase costs such as the installation and maintenance. The expectation that the anchor tension is an objective is linked to the limits when it serves as a constraint.

Since the anchor tension was constructed as a constraint in the PSO, it has a limit, and any tension larger than the limit were penalised and any tension that was smaller than the limits were deemed as constraint compliance. In the MOPSO, the anchor tension is the second objective. Instead of claiming the smaller the better, the 'origin' of the second objective is to set the anchor tension at 10% smaller than the limit recorded in section 6.6.2. For the purpose of consistency, the second objective of the mooring problem has been normalised to the anchor tension limit in section 6.6.2.

7.4.2 MOPSO constraints and penalty functions

The penalty scheme was kept the same as in the section 7.3.2, but without the penalty component from the anchor tension. Because this penalty component has been moved to the objectives group, the penalty scheme only has three parts. The penalty threshold *k* factor is the same as the Group I (LTP group) listed in Table 7.1 because of the comparative results from PSO where only Group I (LTP group) had optimised solutions.

7.4.3 MOPSO results and discussion

The results extracted from the MOPSO of the six runs are summarised in Table 7.3, with an extra column showing the comparisons of the reference mooring system before optimisation. As with the PSO for the first objective, a good performance of the first objective has a low fitness (less than 1) value and no constraint violations. But in terms of the second objective, since it has the smallest normalised value of 0.9

(90% of the anchor tension limit) instead of 0, a good performance of the second objective has a fitness value in the range of 0.9 to 1 and no constraint violations.

It is clear from Table 7.3 that five runs out of six met the criteria of optimisation discussed above. Even trial No.4 has the second objective slightly larger than one (1.001), all the constraints fully complied, hence, strictly speaking, except for Run No. 4, all the remaining solutions are improvements to the reference mooring system.

Figure 7.5 shows both objectives versus iterations, but instead of showing a gradually decreasing curve, the fluctuating curve in blue indicates the Normalised Material Cost (NMC) of the system. The green curve shows the second objective value, Normalised Anchor Tension (NAT), with its corresponding *y* axis on the right hand side. A close observation of the pattern of these two curves shows that a convex in one curve is associated with a concave. This represents the 'trade-off' of one objective to another in the Pareto optimal selection. It is clear that the chance of finding optimised results is enhanced compared to the PSO.

Figure 7.6 represents the Normalised Penalty value (NP) versus iteration. It is clear that for all six runs, the final penalty values at the end of the 100^{th} iteration are zero. Candidates with penalty values during iteration are not feasible in terms of mechanical failure or system failure. The penalty scheme was set for the purpose of optimisation, and therefore those candidates with penalty values are inferior to the reference case in that particular criterion. All the penalty values have been normalised to the *k* factor of Group I (LTP group) in Table 7.1. As with the fluctuating pattern of the PSO, the penalty scheme shows great ability at leading the algorithm in the direction of improvement. The abrupt change of the pattern is due to the large penalty threshold associated with the scheme.

Table 7.3 MOPSO results extraction

Run No.		**1**	**2**	**3**	**4**	**5**	**6**	**REF CASE**
Final Solutions		{128,91,364, 111,142,70}*****	{123,94,364, 87,147,73}*****	{118,103,362, 90,120,70}*****	{129,91,360, 100,130,73}*****	{128,102,353, 87,142,70}*****	{128,104,351, 76,177,70}*****	{125,100,360, 116,152,108}*****
Obj 1: NMC		0.835	0.618	0.922	0.579	0.844	0.698	1.000
Obj 2: NAT		0.998	1.000	0.994	1.001	0.984	0.981	1.000
EV	Constraints violated	N	N	N	N	N	N	N/A
	Saving in %[#]	0.97	0.24	0.82	0.34	0.23	0.33	0
	Violation in %[#]	0	0	0	0	0	0	0
PV	Constraints violated	N	N	N	N	N	N	N/A
	Saving in %[#]	7.9	25.8	5.0	20.0	25.9	27.4	0
	Violation in %[#]	0	0	0	0	0	0	0
BV	Constraints violated	N	N	N	N	N	N	N/A
	Saving in %[#]	49.3	24.1	41.0	18.0	38.9	11.8	0
	Violation in %[#]	0	0	0	0	0	0	0

Note: REF CASE, is the mooring system before optimisation
NMC, normalised material cost
NAT, normalised anchor tension
EV, PV, BV and ***** are the same as in Table 7.2
N, stands for non-violated
[#], the percentage is compared to the corresponding value in the mooring system before optimisation

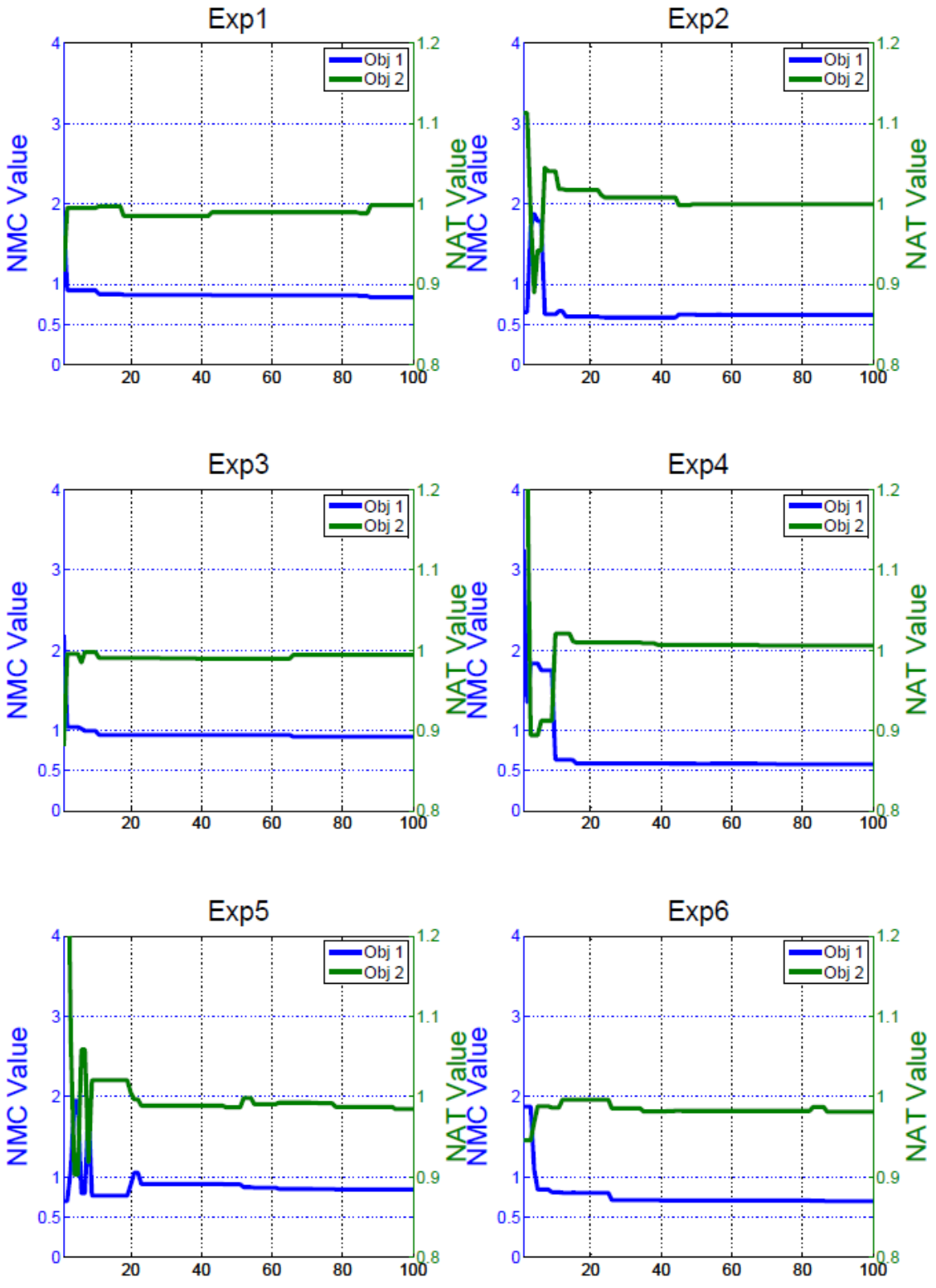

Figure 7.5 Normalised objective 1 and 2 versus iteration

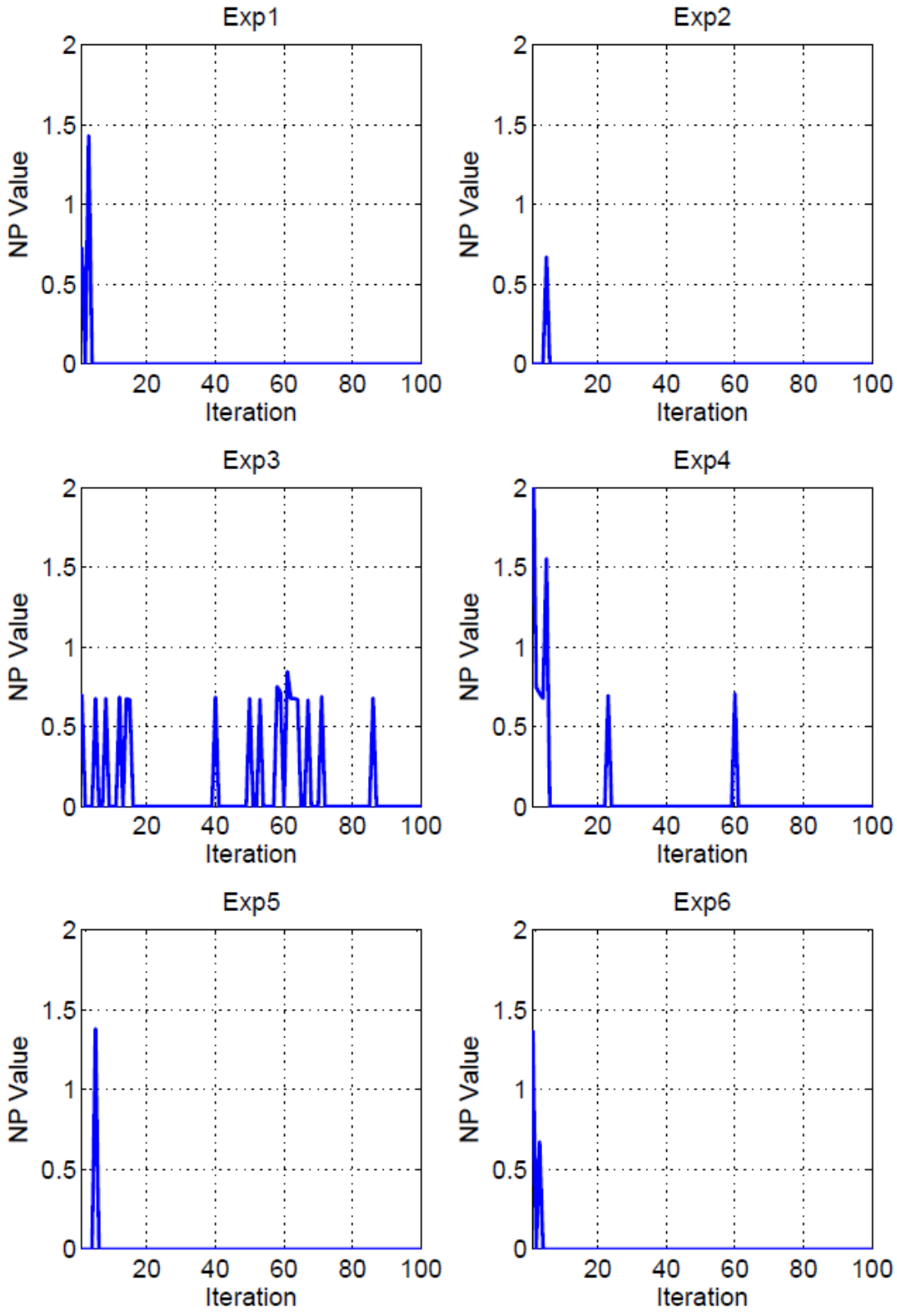

Figure 7.6 Normalised penalty value versus iteration

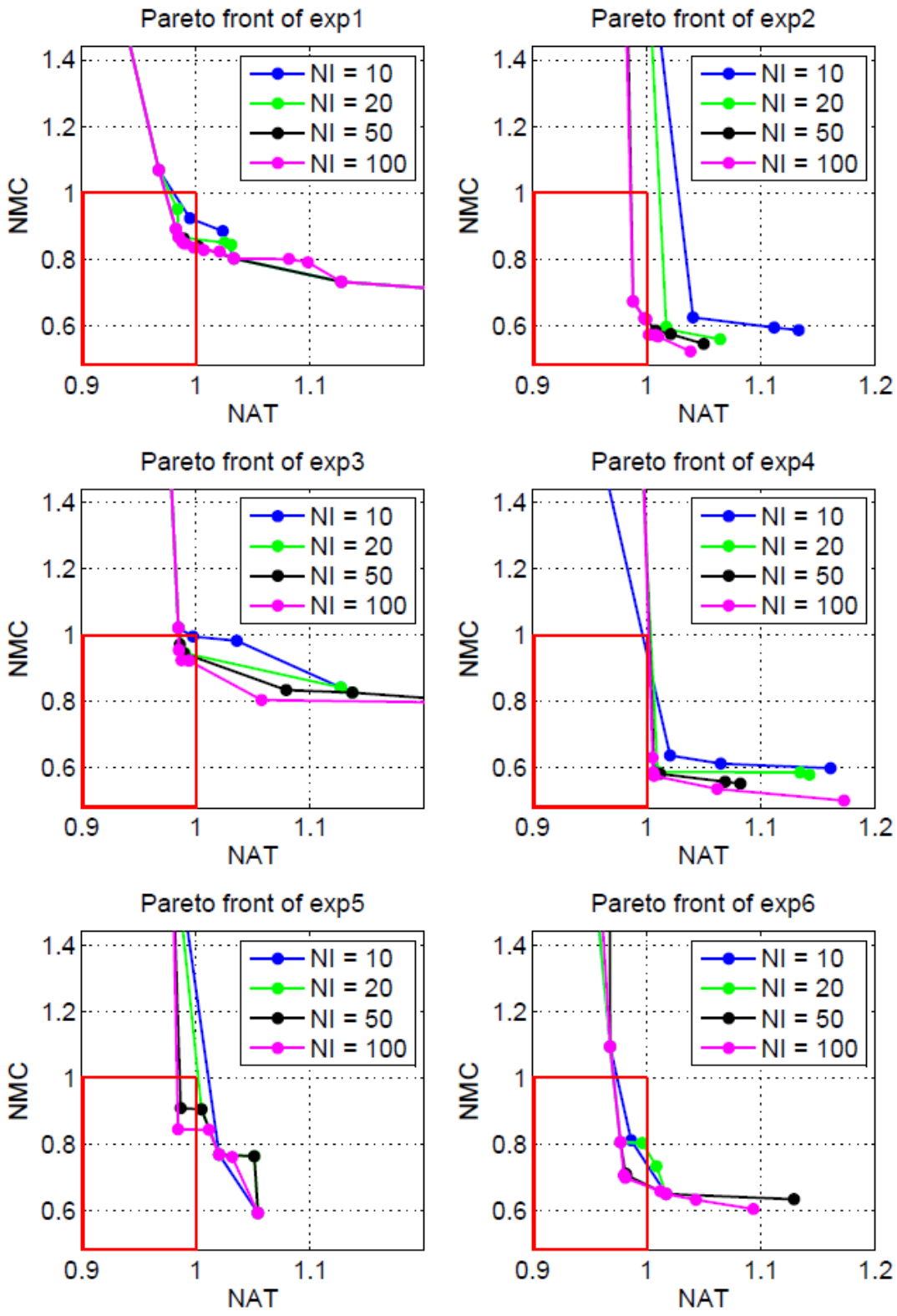

Figure 7.7 Pareto front curves with different iteration numbers

The convergence of the MOPSO can be evaluated with the Pareto front curve (as shown in Figure 7.7), where NI stands for the number of iterations. The rigid red rectangle in the diagram represents the improvement zone, which means any particles located in this region are better than the reference mooring system. It is clear that the Pareto frontier curve is approaching the improvement zone with the increasing iterations. In other words, the particles are guided to the expected directions with the increase of iterations. Even though the performance is improving as the iterations increase, it is unwise, computationally, to let the algorithm run forever. Meanwhile, in terms of simulation efficiency, out of six runs, five of them have found improved solutions within 50 iterations.

7.4.4 MOPSO & PSO comparison, performance based design and optimisation studies

A total of six runs give six different mooring configurations, and five of them are deemed to be improvements on the reference case – i.e. to have been optimised. If the best configuration needs to be selected from the five optimised solutions, the selection scheme introduced in section 6.3.4 can be used to make the ultimate decision. In contrast, the PSO gave only two improved solutions in Group I (LTP group), whereas Group II and III (MTP and STP group) failed to fulfil the purpose of optimisation. Meanwhile, design engineers have a tool that generates multiple design schemes with a range of performances.

7.4.4.1 Solution comparison

When comparing the effectiveness and efficiency of PSO and MOPSO, the latter demonstrated greater capacity and reliability. The behaviour of PSO in Figure 7.4 (Group I curve in red) depicts a fluctuating, and therefore an unhealthy curve in terms of convergence. Even though it came out with optimised solutions, the possibility of finding the results was small and no guarantee can be made based on the algorithm. Finding the optimised solution for this mooring case may depend on the quality of the initial populations. Superior initial populations can lead to optimised results, while inferior initial populations may result in divergence.

Figure 7.7, by comparison, shows the Pareto front curve with the increasing iterations. The approaching trend of the Pareto front that curves to the ‘origin’

demonstrated a healthy convergence by the MOPSO. Furthermore, all of the runs that come up with optimised candidates have found solutions within 50 iterations. Therefore, it is obvious that MOPSO demonstrated greater ability in terms of both effectiveness and efficiency.

7.4.4.2 Performance based results selection

Technically speaking, all the particles on the Pareto front curve, located inside the rectangle shown in Figure 7.7, are improved candidates compared to the mooring reference case. Therefore, if all the complied and improved particles are summarised together, more than five candidates have qualified. Compared with the PSO, where the quality of a particle can be simply calculated based on a single objective value, the selection of multi-objective particles were based more or less on a 'trade-off' process. When different importance factors applied to different objectives, the selection of qualified candidates may involve a high level of intelligence, knowledge, and experience.

To use the mooring case study here as an example, the second objective was adopted from one of the constraints in the PSO to avoid conflicting and competing. Furthermore, as described in section 6.3, the selection of the 'origin' in MOPSO in the mooring optimisation is to assist the algorithm to find improvements. This 'origin' is only a reference point that depends on an engineers' judgement. All the improved solutions from MOPSO can be further examined with all the relevant information.

First of all, examine the result from run No.4 which was categorised as 'constraint violated' beforehand. If this was found in the PSO, the solution would be filtered out for selection. However, in MOPSO, it provides engineers and management with all sorts of figures and values, as shown in Table 7.3, so this solution can still be discussed. This solution gives the smallest material costs. Compared to the second smallest material costs in run No.2, it has about 7% of material saving, with only 0.5% of anchor tension violation, which may differ due to different simulation packages. If the scenario that improves the strength of the anchor by 0.5% is more economically friendly and easier to implement than the 7% cost of material, this is most likely solution to be selected.

Likewise, compared with runs No.4 and 6, about 20% extra material has been added on in the configuration shown in run No.6. However, if anchor tension limitations are strictly carried out, and management found transport to the offshore locale to be important, run No. 6 showed a 27% saving of the payload in the mooring system that would increase the storage of FPSO to a great extent.

If safety is the first priority, then selecting run No.2 as the solution would add an extra 50% of strength to the mooring leg and increase the designer's confidence. The safety factor is also doubled compared to the reference case. In all, the discretion of all potentially optimised results relies fully on which are the preferred priorities in the project.

MOPSO provides different sets of solutions, although the benefits of all these solutions are different. No matter which condition is the most important, the performance based MOPSO can provide enough potential to optimise the design with different performances. MOPSO gives a fully flexible ability to accommodate all the needs. Furthermore, apart from being able to search multiple objectives in the optimisation, other relevant information provided in the algorithm enables an interaction to take place between the results, engineers, and management.

7.5 Chapter summary

This chapter has presented an industry based mooring case study which was selected for optimisation under conflicting and competing constraints. This practical case revealed the complexity of offshore optimisation and the limitations of stochastic search algorithms. With a relatively large number of constraints to satisfy, the penalty scheme reflected its drawbacks in recognising superior and inferior solution candidates. A study based on the penalty functions reveals that the quality of the solution is heavily related to the quality of the penalty functions. The boundary of the improvement zone is blurred with an inferior defined penalty functions. Only superior penalty functions can help PSO to draw the border. The correctness of using penalty functions is based on trial and errors, but non-conforming results from the PSO do not suggest that a single objective optimisation is suitable. Instead, it fully

demonstrated that using a penalty scheme can lead the search astray and can guide the swarm in the wrong direction if the settings are inappropriate. Many effects may be needed to tune up the penalty scheme to fit every individual problem. Therefore, in problems of a highly constrained category, conflicting constraints criteria make it increasingly difficult to find an appropriate penalty scheme.

This chapter has demonstrated the ability of MOPSO software under the new design philosophy of designing and optimising marine mooring structures. One of the findings suggested that the multi-objective optimisations could provide enough information for a performance based design. Furthermore, the ability to supply multiple solutions sets the server of a highly constrained offshore application in a high quality mode.

The mooring case has been studied with both PSO and performance based MOPSO. The results found that with the aid of MOPSO, the searching ability of the algorithm has been greatly enhanced. Meanwhile, the chances of finding feasible solutions have been largely increased with the same terms and conditions. Apart from all the new benefits, the choices of selecting final solutions have been expanded by the introduction of an external repository that recorded all the Pareto optimal sets. This type of flexible optimisation searching helps promoting performance in future designs.

8 CONCLUSIONS & REFLECTIONS

8.1 Thesis summary

The fulfilment of this thesis was undertaken in a joint collaboration between AMOG Consulting Pty Ltd., Melbourne office and UoW Australia under the doctoral scheme. This research involves academic research and tuition, industrial placements as well as professional software training. All the achievements have been documented in this thesis, and a summary is given in the following.

Within the aim of optimising offshore mooring design, this project has examined the methodologies, software package, and tools that are used throughout offshore design and analysis. It began with a review of offshore infrastructures that covered offshore platforms and mooring systems. This provided sufficient background information about oil exploration in terms of equipment and technology. This was followed by an investigation of mooring line analysis, design standards, and processes, from which an improvement to the flexibility iteration approach has been developed in the analysis of offshore mooring lines.

Search and optimisation techniques have been reviewed to facilitate the quality of automation and solutions to offshore engineering applications. The features and characteristics of different search methods have been investigated. Traditional search methods proved to be efficient in continuous and smooth behaved functions, while artificial intelligence search methods work as an enhanced version of traditional optimisation, in that they are more advanced in searching for global optimisation, even in discrete and non-differentiable functions.

As an optimisation technique gaining increasing popularity due to its fast speed of convergence and relatively easy coding in programming, PSO has been selected as a suitable technique for optimising offshore cables, and its mathematical characteristics and features with regards to its convergence speed and stability have been studied. As a comparison, the genetic algorithm (GA), which is another evolutionary optimisation scheme, has been benchmarked against PSO. The results proved that both GA and PSO have a fast speed of convergence, but within the same

number of computational runs, PSO performed better and with a faster running time than GA in an engineering application.

Even with a sophisticated simulation package and optimisation technique, searching for feasible solutions to a mooring problem can still be difficult. The case study undertaken in Chapter 7 demonstrated a highly constrained mooring problem that was ultimately solved by an evolutionary design software package that interfaced with OrcaFlex. The results from a single objective PSO showed the competing constraints that have guided the whole system in an unfavourable manner because all the constraints were lumped into one penalty function, so the quality of candidates cannot easily be distinguished or quantified when different criteria have been violated. Therefore, a more sophisticated method is required under this circumstance.

A new performance based design method was developed to tackle the problems experienced in single objective PSO optimisation. It required the assistance of multiple objectives to reduce the number of constraints that are either conflicting or competing with a problem. The new design method gives engineers multiple advisable solutions with different performances. This new design method expanded potential advisable solutions through an external repository in the evolutionary software. 'Trade off' decisions can then be made by investors according to the different performances of the potential candidates. The software developed used the Pareto global optimum selection technique with improved customising features.

A case study has been used to demonstrate the difficulties associated with offshore mooring designs. Within the new design method, the capacity, benefits, and improvements of the evolutionary software have been verified. Compared with the reference case already mentioned, improved solutions were provided with a different performance in terms of anchor tension, payload, material cost, and the restoring force.

The current work undertaken has developed an evolutionary software package with a performance based design method that can provide offshore engineers with more

advisable solutions for different performances. The optimisation software has been successfully tested by using a mooring design.

8.2 Original contributions

An examination of offshore mooring design, analysis and methodology, and improvements to the methods and tools has been included in the research of this project. The publication 'Taut-slack Algorithm for Analysing the Geometric Nonlinearity of Cable Structures' [141] improved the flexibility iteration analysis approach and demonstrated its potential benefits with a case study. This project has presented issues and concerns related to the design of offshore mooring structures. An appropriate method for optimising mooring structures, particle swarm optimisation (PSO), was selected from a thorough review of current optimisation techniques and benchmarking. A publication based on a comparison between genetic algorithm and particle swarm optimisation examined the performance and efficiency of these two evolutionary optimisation algorithms [142]. The application of MOPSO in offshore mooring engineering was the first of its kind. Evolutionary offshore mooring design and optimisation software was built on the basis of the multiple objective particle swarm optimisation.

This research has developed software for external control mooring design and optimisation with a performance based design method. This is the first of its kind in performance based design in offshore engineering. The new method developed in this project involved the software using an external repository to store all the advisable solutions for further processing by engineers on the basis of cases. This new method can provide more potential solutions that have been either neglected or penalised in normal optimisation design methods, and also provides offshore cable design problems with more lenient and flexible constraints, because the problems no longer required careful examination at the initial design stage. Therefore this method and this software give offshore engineers a superior tool for the design and optimisation of offshore mooring systems. The potential benefits to the offshore industry can now readily be foreseen.

8.3 Review of achievements

In Chapter One, the introduction of the thesis, a few milestones with regards to aims and objectives were set at the beginning. To examine the achievements of this project in a tangible manner, Table 8.1 summarises a list of objectives set out in Chapter 1 and provides evidence that the objectives have been achieved.

Table 8.1 Review of achievement

Objectives	Evidence of achievements
Familiarisation of the offshore terminology, design concepts, codes and practice	• Chapter 3, section 3.2 and 3.3 • Chapter 3 'mooring design and analysis' section 3.10 • Working experience gained during placements
Understanding the methodologies of offshore mooring design and analysis	• Chapter 3 'mooring design and analysis', section 3.7, 3.8 and 3.9 • Publication and presentation • Working experience gained during placements
Investigation of mooring line behaviour for the mooring system	• Chapter 3 'mooring design and analysis', section 3.5 and 3.6 • Publication and presentation
Obtaining a suitable case study of mooring design	• Working experience gained during placements • Chapter 3 'mooring design and analysis', section 3.11 • Software illustrated in Chapter 6
Examination of the technologies of optimisation and locating the appropriate application of optimisation	• Chapter 4 'optimisation techniques' • Chapter 5 'particle swarm optimisation' • Chapter 6 'benchmarking of optimisation algorithms' • Publication and presentation
Developing new tools and	• Chapter 6 'MOPSO' on mooring system'

methodologies for the design of mooring systems	• Software illustrated in Chapter 6
Demonstrating and proving the capabilities and potential improvements over existing practice of any new tools and methodologies developed	• Chapter 7 'PSO on offshore mooring system' • Chapter 6 'MOPSO' on mooring system' • Software tools and methodologies illustrated in Chapter 6
Coding capacity and programming of software packages which fulfilled the optimisation and stimulation	• Software illustrated in Chapter 6 • Training obtained in Matlab • Training obtained in Orcina • Publication and presentation

8.4 Future work recommendations

Due to the time limit set for a doctorate scheme, it is impossible to cover every aspect of the scope of the research project, but the proposed performance based design method for mooring MOPSO can serve as the first step to further works.

8.4.1 Further improvement of offshore design method

As recommendations for further work, the software can be expanded to accommodate an offshore system that simultaneously includes both moorings and risers, while this coupled system of the mooring riser and vessel can be analysed iteratively. This can be fulfilled by modifying the software to communicate with other third party software packages such as Ariane 7. The OrcaFlex can construct a complex coupled system, but a running fully coupled dynamic analysis of a whole offshore system including the platform, mooring, and risers, can be time consuming. This is not considered efficient enough in design. Instead of changing the industrial design circle, the interaction between the mooring constraints to riser design can be developed through a feedback system. The results from a fast frequency domain analysis of mooring provided by Ariane 7 can be collected and then processed to OrcaFlex. The interaction of the offshore system can be further optimised by considering both the riser and mooring system. Within the framework of the

performance based design, relevant aspects of interests can be considered with different levels of priority.

This would change the current method for designing offshore systems since the original distinctive boundary between riser and mooring design can be joined at the stage of conceptual design. The risers and moorings can be designed simultaneously so that the offset limits set by designing the risers to the mooring system can be more flexible. This involves a re-design, as shown in Figure 8.1. Within the consideration of designing a mooring, the offset of the vessel that is often set by the riser design can be beneficial to both parties in that the mooring system can be designed in a more rational way.

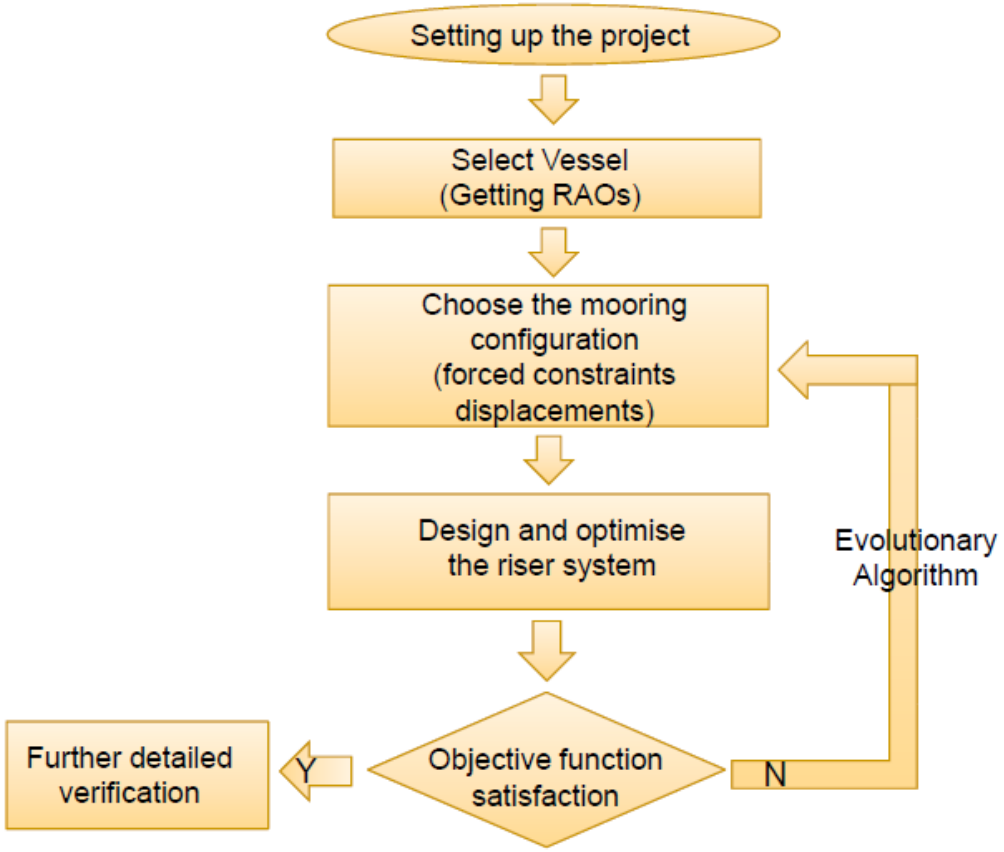

Figure 8.1 The new offshore design philosophy

8.4.2 Software application recommendation

The evolutionary software developed here integrated the evolutionary search algorithm with one of the commercial simulation package OrcaFlex. In order to fulfil

the tasks of offshore system integration design, complex and computationally expensive problems are expected, and therefore more sophisticated software is required to communicate with other commercial packages.

Advantages such as parallel computing, which has already been embedded in the latest version of OrcaFlex, can be taken into account when interfacing Matlab or another coding language such as Python. An enhanced computational capacity can improve the response performance and decrease the computational time, and so different load cases can be verified more efficiently. The increase of computational power would reinforce the application of the optimisation algorithm to larger and more complex offshore projects.

In the current optimisation project, the software verified the most extreme case with an assumption made by the author, but in practice, a number of loadcases should be considered during the time of evaluation. It is proposed that the system should include the loadcases simulation in a robust manner that would allow users to specify the consequence of loadcases based on their level of importance level and engineering judgement.

The frequent use of OrcaFlex is considered to be expensive, so it is suggested that a neural network be trained with the assistance of the performance based design method proposed.

8.4.3 Further offshore mooring risk management study

The author has found that the trend of searching for economical optimum solutions has been gradually overtaken by minimising offshore risks through internship and conversation with offshore engineers. Canonical optimisation tends to give a determined solution that can only fit the specific purpose nominated by clients. The proposed performance based design in this research gives more indeterminate solutions with different performances which explores the solution pool in a way that cannot be predicted, so the potential for providing flexible solutions has been enhanced.

To fit the framework of risk management in offshore mooring design, further identification, assessment, and uncertainty of risks can be analysed based on the results obtained from performance based design. For example, the mooring material was used to trade off the payload, anchor tension, restoring force and breaking limit of the mooring for some of the solutions in the case illustrated in Chapter 6. That is, if different performances are associated with risk levels through risk management techniques, such that a sophisticated method for analysing offshore mooring risks can easily be developed. It is suggested that a different factor of safety be linked to the candidates with different performance.

The customised external expository in the performance based design is able to store a set of potential solutions, so a multiple solution pool can be obtained within a single running of the optimisation. As such, a more reliable mooring system in terms of risks can be found with a minimal amount of effort in searching for different performance solutions.

9 REFERENCES

1. IMF, *World Economic Outlook - Tensions from the two-speed recovery unemployment, commodities, and captial flows*. 2011.
2. RET, *Australia's Offshore Petroleum Industry, Fact Sheet 1*. 2010.
3. *Oil production and consumption*. 2012 [cited 2012 01/02]; Available from: http://www.economist.com/blogs/dailychart/2011/06/oil-production-and-consumption.
4. *The world factbook*. 2012 [cited 2012 01/02]; Available from: https://www.cia.gov/library/publications/the-world-factbook/geos/xx.html.
5. *2010 in review*. 2012 [cited 2012 01/02]; Available from: http://www.bp.com/sectiongenericarticle800.do?categoryId=9037145&contentId=7068556.
6. BP, *BP statistical review of world energy* 2011, BP.
7. Dawson, T.H., *Offshore structural engineering*. 1983, Englewood Cliffs, New Jersey: Prentice-Hall, INC.
8. *Offshore Moorings*. 2012 [cited 2012 10/05/2012]; Available from: http://www.offshoremoorings.org/.
9. Cunliffe, N., et al., *Evolutionary Design of Marine Riser Systems*, in *International Conference on Offshore Mechanics and Arctic Engineering*. 2004: Canada.
10. Sarker, R., M. Mohammadian, and X. Yao, *Evolutionary Optimization*. 2002: Kluwer academic publishers.
11. Michalewicz, Z. and D.B. Fogel, *How to solve it: modern heuristics*. 2000, Berlin: Springer.
12. Hardy, S. and J. Ramjeet, *Reflections on how to write and organise a research thesis.* Nurse researcher, 2005. **13**(2): p. 27-39.
13. API RP-2SK, *Recommended practice for design and analysis of stationkeeping systems for floating structures.* 2005.
14. DnV OS-E301, *Position Mooring*. 2001, Det Norske Vertias.
15. Rajasekar, S., P. Philominathan, and V. Chinnathambi, *Research methodology*. 2006, School of Physics, Bharathidasan University: Tamilnadu, India.
16. Wasonga, T.A. and J.F. Murphy, *Learning from tacit knowledge: the impact of the intership.* The International Journal of Educational Management, 2006. **20**(2): p. 153-63.
17. Stanton, M., *Interships: Learning by Doing: Current Supplement to Occupational Outlook Handbook.* Occupational outlook Quarterly, 1992. **36**(2): p. 30-33.
18. Krathwohl, D.R. and L.W. Anderson, *A taxonomy for learning, teaching and assessing: a revision of Bloom's taxonomy of educational objectives*. 2001, New York: Longman.
19. Fenwick, A., *Structural topmisation using genetic algorithms*, in *Faculty of engineering*. 2006, University of Wollongong.
20. Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*. 1994, New York: Springer-Verlag.

21. Beighlter, C.S., D.T. Phillips, and D.J. Wilde, *Foundations of optimisation*. 1979, New Jersey: Prentice-Hall.
22. Chapra, S.C., *Applied numerical methods with Matlab for engineers and scientists*. 2005, New York: McGraw Hill.
23. Arora, J.S., *Introduction to optimisation desgin*. 1989, New York: McGraw Hill.
24. Onwubiko, C., *Introduction to engineering design optimisation*. 2000, New Jersey: Pretice-Hall.
25. Michalewice, Z. and D.B. Fogel, *How to solve it: modern heuristics*. 2000, Berlin: Springer.
26. Glover, F., *Tabu search -- Part I*. ORSA Journal on computing, 1989. **1**(3): p. 190-206.
27. Glover, F., *Tabu search -- Part II*. ORSA Journal on computing, 1990. **2**(1): p. 4-32.
28. Glover, F. and M. Laguna, *Tabu search*. 1997, Kluwer, Norwell: Springer.
29. Shapiro, A.F., *The merging of neural networks, fuzzy logic, and genetic algorithms.* Insuracne: Mathematics and Economics, 2002. **31**: p. 115-31.
30. Guarize, R., et al., *Neural networks in the dynamic response analysis of slender marine structures.* Applied Ocean Research, 2008. **29**: p. 191-98.
31. Rafiq, M.Y., G. Bugmann, and D.J. Easterbrook, *Neural network design for engineering applications.* Computers and Structures, 2001. **79**(17): p. 1541-52.
32. Pham, D.T. and D. Karaboga, *Intelligent optimisation techniques*. 2000, London: Springer.
33. Metropolis, N., et al., *Equations of state calculations by fast computing mechines.* Journal of Chemical Physics, 1953. **21**: p. 1087-92.
34. Kirkpatrick, S., C. Gelatt, and M. Vecchi, *Optimiztion by simulated annealing.* Science, 1983. **220**(4598): p. 671-80.
35. van Laarhoven, P.J.M. and E.H.L. Aarts, *Simulated annealing: theory and application*. 1987, Norwell, MA: Springer.
36. Preux, P. and E.-G. Taibi, *Towards hybrid evolutionary algorithms.* International transactions in operational research, 1999. **6**(6): p. 557-70.
37. Burke, E.K. and A.J. Smith, *Hybrid evolutionary techniques for the maintenance scheduling problem.* IEEE Trans Power Syst, 2000. **15**(1): p. 122-8.
38. Cunliffe, N., C. Baxter, and T. McCarthy, *Evolutionary design of marine riser systems*, in *23rd International conference on offshore mechanics and arctic engineering*. 2004: Vancouver, Canada.
39. Galinier, P. and J.K. Hao, *Hybrid evolutionary algorithms for graph coloring.* J Comb Optim, 1999. **3**(4): p. 379-97.
40. Premalatha, K. and A.M. Natarajan, *Combined heuristic optimization techniques for global minimization.* Int. J. Advance. Soft Comput. Appl., 2010. **2**(1): p. 85-99.
41. Schmidt, H. and G. Thierauf, *A combined heuristic optimization technique.* Advances in engineering software, 2003. **36**: p. 11-19.
42. Chakrabarti, S.K., *Handbook of offshore engineering*. Vol. 1. 2005: Elsevier Ltd.
43. Ulbricht, W.R., et al. *Design, fabrication and installation of the Anger TLP foundation system*. in *The 26th Annual Offshore Technology Conference (OTC7626)*. 1994. Houston, USA.

44. Kyriakides, S. and E. Corona, *Mechanics of Offshore Pipelines Vol. I Buckling and Collapse* 1ed. Vol. 1. 2007: Elsevier Ltd.
45. Lyle, D., *hoover/Diana's record performance.* Oil & Gas Investor, 2002. **ExxonMobil's Hoover/Diana - A Deepwater Pioneer**: p. 44-45.
46. Kenney, J.J., et al. *Na Kika hull design interface management challenges and successes*. in *The offshore Technology conference*. 2004. Houston, USA.
47. Portella, et al. *DICAS Mooring Systems: Practical Design Experience to Dismystify the Concept*. in *OTC*. 2003. Houston, USA.
48. Shimamura, Y., *FPSO in Australia and New Zealand, Status and Future*. 2009, WA Oil and Gas Facilities.
49. API-RP-2SK, *Design and Analysis of Stationkeeping Systems for Floating Structures*. 2005.
50. ISO-19901-7, *Stationkeeping systems for floating offshore structures and mobile offshore units*. 2005.
51. Mombaerts, J., *Mooring course for ARIANE users*. 2006, Bureau Veritas.
52. DNV OS-C101, *Offshore Steel Structures (LRFD)*. 2004.
53. Orcina. *Home of OrcaFlex*. 2010 [cited 2010 28/10/2010]; Available from: http://www.orcina.com/.
54. Orcina, *OrcaFlex help manual*. 2005.
55. Irvine, H.M., *Cable Structures*. 1992, London: The MIT Press.
56. Lewis, W., M. Jones, and K. Rushton, *Dynamics Relaxation Analysis of the Non-linear Response of Pre-tensioned Cable Roofs.* Computers & Structures, 1984. **18**: p. 989-997.
57. Krishna, P., *Cable-suspended Roofs*. 1978, New York: McGraw-Hill.
58. Peyrot, A.H. and A.M. Goulois, *Analysis of Cable Structures.* Computers & Structures, 1979. **10**(5): p. 805-813.
59. Silva, R.M.C. and C.E. Parente, *Review of design criteria for deepwater risers and mooring systems in a multidirectional environment*, in *24th Int Conf on Offshore Mech and Arct Eng.* 2000.
60. O'Brien, W.T. and A.J. Francis, *Cable movements under two-dimensional loads.* J. Struct. Div. ASCE, 1964. **89**: p. 89-123.
61. O'Brien, W.T., *General solution of suspended cable problems.* J. Strucut. Div. ASCE, 1967. **93**: p. 1-26.
62. Jaymaraman, H.B. and W.C. Knudson, *A curved element for the analysis of cable structures.* Computers and Structures, 1981. **14**: p. 325-33.
63. Huang, T. *A static equilibrium formulation including axial deformation for marine cables*. in *2nd Int Offshore and Polar Eng Conf (ISOPE)*. 1992. San Francisco, USA.
64. Chucheepsakul, S. and T. Huang. *Effect of axial deformation on the equilibrium configruation of marine cables*. in *5th Int Offshore and Polar Eng Conf (ISOPE)*. 1995. Hague, Netherlands.
65. Karoumi, R., *Some modelling aspects in the non linear finite element analysis of cable suspported bridges.* Computers & Structures, 1998. **71**: p. 397-412.
66. Andreu, A., L. Gil, and P.A. Roca, *A new deformable catenary element for the analysis of cable net structures.* Computers & Structures, 2006. **84**: p. 1882-1890.
67. Ormberg, H., N. Sodahl, and O. Steinkjer. *Efficient analysis of mooring systems using de-coupled and coupled analysis*. in *Proceedings of the 17th International Conference on Offshore Mechanics and Arctic Engineering (OMAE)*. 1998.

68. Wang, L., V. Hansen, and E. Katla. *Independent verification of deepwater SCR designs*. in *Offshore Technology Conference (OTC)*. 2006.
69. Rodrigues, M.V., et al. *Steel catenary riser design based on coupled analysis methodology*. in *Proceedings of the 27th International Conference on Offshore Mechanics and Arctic Engineering (OMAE)*. 2008. Estoril, Portugal.
70. DNV OS-C201, *Structural design of offshore units (WSD method)*. 2011.
71. Chen, P., S. Chai, and J. Ma, *Performance evaluations of taut-wire mooring systems for deepwater semi-submersible platform*, in *30th International Conference on Ocean, Offshore and Arctic Engineering, OMAE2011*. 2011: Rotterdam.
72. Low, Y.M. and R.S. Langley, *A hybrid time/frequency domain approach for efficient coupled analysis of vessel/mooring/riser dynamics.* Ocean Engineering, 2008. **35**(5-6): p. 433-446.
73. Low, Y.M. and R.S. Langley, *Time and frequency domain coupled analysis of deepwater floating production systems.* Applied Ocean Research, 2006. **28**(6): p. 371-385.
74. Guarize, R., et al., *Neural networks in the dynamic response analysis of slender marine structures.* Applied Ocean Research, 2008. **In Press, Corrected Proof**.
75. API-RP-2SM, *Recommended Practice for Design, Manufacturing, and Maintenance of Synthetic Fiber Ropes for Offshore Mooring.* 2007.
76. DNV-RP-C205, *Environmental Conditions*. 2010.
77. Marcollo, H. and A. Eassom, *Optimisation case study*. 2011, AMOG: Melbourne.
78. SOFEC. *SOFEC External Turret Mooring Systems*. 2012 [cited 2012 27/05]; Available from: http://www.sofec.com/MooringSystems/ExternalTurretMoorings.aspx.
79. Kennedy, J. and R. Eberhart, *Particel swarm optimization.* IEEE international conference on neural networks, 1995. **IV**: p. 1942-8.
80. Kathiravan, R. and R. Ganguli, *Strength design of composite beam using gradient and particle swarm optimization.* Composite structures, 2007. **81**(2007): p. 471-9.
81. Coello, C. and C. Luna, *Use of particle swarm optimization to design combinational logic circuits*, in *5th International conference on evolvable systems: from biology to hardware, ICES 2003*. 2003.
82. Zheng, Y., et al., *Robust pid controller design using particle swarm optmizer*, in *IEEE international symposium on intelligence control*. 2003. p. 974-9.
83. Krohling, R., L. Coelho, and Y. Shi, *Cooperative particle swarm optimization for robust control system design*, in *7th Online world conference on soft computing in industrial applications*. 2002.
84. McCluskey, S., *Application of particle swarm optimisation to reinforced concrete beam design*, in *Faculty of engineering*. 2008, University of Wollongong.
85. Hassan, R., et al., *A comparison of particle swarm optimization and the genetic algorithm*, in *1st AIAA multidisciplinary design optimization speclialist conference*. 2005: Austin, TX.
86. Shi, Y. and R. Eberhart, *A modified particle swarm optimizer.*, in *IEEE international conference on evolutionary computation*. 1998: Piscataway, NJ.
87. Goldberg, D., *Genetic algorithms in search, optimization, and machine learning.* 1989, New York: Addison-Wesley.

88. Gen, M. and R. Cheng, *Genetic algorithms and engineering design*. 1997, New York: Wiley-Interscience.
89. Reeves, C.R. and J.E. Rowe, *Genetic algorithms - principles and perspectives*. 2003, London: Kluwer academic publishers.
90. Tanaka, R.L. and C.A. Martins, *parallel dynamic optimisation of steel risers.* Journal of offshore mechanics and arctic engineering, 2011. **133**(1): p. 11302-9.
91. Bäck, T., D.B. Fogel, and Z. Michalewice, *Evolutionary computation 1: Basic algorithms and operators*. 2000, Bristol: Institue of physics publishing.
92. Whiteley, D., *A genetic algorithm tutorial.* Statistics and computing, 1994. **4**: p. 65-85.
93. Perez, R.E. and K. Behdinan, *Particle swarm approach for structural design optimization.* Computers and Structures, 2007. **85**(2007): p. 1579-88.
94. Brits, R., A.P. Engelbrecht, and F. Van den Bergh, *A Niching Particle Swarm Optimisation*, in *Fourth Asia-Pacific Conference on Simulated Evolution and Learning*. 2002.
95. Van den Bergh, F. and A.P. Engelbrecht, *Effects of Swarm Size on Cooperatie Particle Swarm Optimiser*, in *Genetic and Evolutionary Computation Conference*. 2001.
96. Eberhart, R. and Y. Shi, *Comparing inertia weights and construction factors in particle swarm optimization*, in *IEEE congress on evolutionary computation (CEC 2000)*. 2000: San Diego, CA. p. 84-8.
97. Ratnaweera, A.C., S.K. Halgamuge, and H.C. Watson, *Particle Swarm Optimiser with Time Varying Acceleration Coefficents*, in *International Conference on Soft Computing in Intelligent Systems*. 2002.
98. Venter, G. and J. Sobieszczanski-Sobieski, *Particle swarm optimization.* AIAA, 2003. **41**(8): p. 1583-9.
99. Wang, Y., et al., *Self-adaptive learning based particle swarm optimisation.* Information Sciences, 2011. **181**: p. 4515-38.
100. Liang, J.J., et al., *Comprehensive learning particle swarm optimiser for global optimisation of mutimodal functions.* Evolutionary computation, 2006. **10**(3): p. 281-95.
101. van den Bergh, F. and A.P. Engelbrecht, *IEEE Transactions on A Cooperative Approach to Particle Swarm Optimisation.* Evolutionary computation, 2004. **8**(3): p. 225-39.
102. Hsieh, S.-T. and T.-Y. Sun, *IEEE Transactions on Efficient Population Utilization Strategy for Particle Swarm Optimizer.* Systems, Man, and Cybernetics, 2009. **39**(2): p. 444-56.
103. Mendes, R., J. Kennedy, and J. Neves, *The Fully Informed Particle Swarm: Simpler, Maybe Better.* IEEE Transactions on Evolutionary Computation, 2004. **8**(3): p. 204-10.
104. Venter, G. and J. Sobieszczanski-Sobieski, *Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization.* Struct Multidisc Optim, 2005. **29**: p. 432-44.
105. Kameyama, K., *Particle swarm optimisation - a survey.* IEICE TRANS. INF. & SYST., 2009. **E92-D**(7): p. 1354-361.
106. Di Giampaolo, E., F. Forni, and G. Marrocco, *RFID-Network planning by particle swarm optimization.* Applied computational electromagnetics society journal, 2010. **25**(3): p. 263-72.

107. de Pina, A.A., B.S.L.P. de Lima, and B.P. Jacob, *Tailoring the particle swarm optimization algorithm for the design of offshore oil production risers.* Optimization and engineering, 2011. **12**: p. 215-35.
108. Li, Y.G., W.H. Gui, and C.H. Yang, *Improved PSO algorithm and its application.* Journal of central south university of technology, 2005. **12**: p. 222-26.
109. Kennedy, J. and R.C. Eberhart, *Swarm Intelligence*. 2001, San Mateo, CA: Morgan Kaufmann.
110. Coello Coello, C.A., D.A. Van Veldhuizen, and G.B. Lamout, *Evolutionary Algorithms for Solving Multi-Objective Problems.* 2002: Norwell, MA: Kluwser.
111. Chen, M.-R. and Y.-Z. Lu, *A novel elitist multiobjective optimisation algorithm: Multiobjective extremal optimisation.* European Journal of Operational Research, 2008. **188**: p. 657-51.
112. Boulougouris, E.K. and A.D. Papanikolaou, *Multi-objective optimisation of a floating LNG terminal.* Ocean engineering, 2008. **35**: p. 787-811.
113. Le Huédé, F., et al., *MCS - a new algorithm for multicriteria optimisation in constraint programming.* Ann Oper Res, 2006. **147**: p. 143-74.
114. Knowles, J.D. and D.W. Corne, *Approximating the nondominated Front Using the Pareto Archived Evolution Strategy.* Evolutionary computation, 2000. **8**(2): p. 149-72.
115. Fieldsend, J.E. and S. Singh. *A multi-objective algorithm based upon particle swarm optimisation, an efficient data structure and turbulence*. in *Proc. 2002 U.K. Workshop on Computational Intelligence*. 2002. Birmingham, U.K.
116. Mostaghim, S. and J. Teich. *Strategies for finding good local guides in Muti-Objective Partcile Swarm Optimisation (MOPSO)*. in *Proc. 2003 IEEE Swarm Intelligence Symp*. 2003. Indianapolis, IN.
117. Tripathi, P.K., S. Bandyopadhyay, and S.K. Pal, *Multi-Objective Particle Swarm Optimisation with time variant inertia and acceleration coefficients.* Information Sciences, 2007. **177**: p. 5033-49.
118. Coello Coello, C.A., G.T. Pulido, and M.S. Lechuga, *Handling Multiple Objectives with Particle Swarm Optimisation.* Evolutionary computation, 2004. **8**(3): p. 256-79.
119. Tavakkoli-Moghaddam, R., M. Azarksh, and Sadeghnejad-Barkousaraie.A, *A new hybrid multi-objective Pareto archive PSO algorithm for a bi-objective job shop scheduling problem.* Expert Systems with Applications, 2011. **38**: p. 10812-21.
120. Wang, Y. and Y. Yang, *Particl swarm optimisation with preference order ranking for multi-objective optimisation.* Information Sciences, 2009. **179**: p. 1944-59.
121. Deb, K. and D.E. Goldberg. *An investigation of niche and species formation in genetic function optimisation*. in *Proc. 3rd Int. Conf. Genetic Algorithms*. 1989. San Mateo, CA.
122. Carlson, S.E. and R. Shonkwiler, *Annealing a Genetic Algorithm over Constraints*, in *IEEE. International Conference on SYstem, Man and Cybernetics*. 1998: San Diego, California.
123. Coello, C.A.C., *Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the stae of the art.* Computer Methods in Applied Mechanics and Engineering, 2002. **191**: p. 1245-87.

124. Joines, J. and J. Houck, *On the use of non-stationary peanlty functions to solve nonlinear constrained optimisation problem with GAs.*, in *IEEE Conference on Evolutionary Compuatation*. 1994: Orlando, Florida.
125. Hinterding, R., Z. Michalewicz, and A.E. Eiben, *Adaptation in evolutionary computation: A survey.*, in *IEEE International Conference on Evolutionary Computation*. 1997: Indianapolis, USA.
126. Davis, L., *Bit-climbing, representational bias, and test suite design*, in *International Conference on Genetic Algorithms*. 1991: San Diego, CA.
127. Sooda, K. and T.R. Gopalakrishnan Nair, *A comparative analysis for determining the optimal path using PSO and GA.* International Journal of Computer Applications, 2011. **32**(4): p. 8-12.
128. Yao, X., Y. Liu, and G. Lin, *Evolutionary programming made fast.* IEEE Transactions on Evolutionary Computation, 1999. **3**(2): p. 82-102.
129. Cunliffe, N., *Evolutionary Design of Marine Riser Systems*, in *Department of Civil and Construction Engineering*. 2004, University of Manchester Institute of Science and Technology: Manchester.
130. Wolpert, D.H. and W.G. Macready, *"No free lunch theorems for optimisation:*. Evol. Comput., 1997. **1**: p. 67-82.
131. Galante, M., *Genetic algorithm as an approach to optimize real-word trusses.* Int J Numer Meth Eng, 1996. **39**: p. 361-82.
132. Jingui, L., et al., *An improved strategy for GAs in structural optimisation.* Computers and Structures, 1996. **61**: p. 1185-191.
133. Rajeev, S. and C.S. Krishnamoorthy, *Genetic algorithms-based methodologies for design optimization of trusses.* Journal of structural engineering, 1997. **123**(3): p. 350-8.
134. Leite, L.P.B. and B.H.V. Topping, *Improved genetic operators for structural engineering optimisation.* Advances in engineering software, 1998. **29**: p. 529-62.
135. Sunar, M. and A. Belegunda, *Trust region methods for structural optimization using exact second order sensitiviey.* International journal for numerical methods in engineering, 1991. **32**(2): p. 275-93.
136. Perusco, T., *CO_2 emission optimisation in reinforced concrete beams using particle swarm optimisation*, in *Faculty of engineering*. 2009, University of wollongong.
137. Rajeev, S. and C.S. Krishnamoorthy, *Discrete optimiation of structures using genetic algorithm.* Journal of structural engineering, 1992. **118**: p. 1233-50.
138. Cai, J., *Diskrete optimierung dynamisch belasteter tragwerke mit sequentiellen und parallelen evolutionssatrategien*. 1995, Dissertation an der Universitt-Gesamthochschule-Essen: Essen.
139. Cunliffe, N.D., *Evolutionary design of marine riser systems*, in *Engineering*. 2002, Universy of Machnester Institute of Science and Technology: Manchester, UK.
140. Wong, J., *Matlab-OrcaFlex Guidance*. 2007, Amog Consulting: Melbourne.
141. Wang, Z., T. McCarthy, and M.N. Sheikh. *Taut-slack algorithm for analyzing the geometric nonlinearity of cable structures*. in *International Offshore and Polar Engineering Conference (ISOPE)*. 2011. Maui, Hawaii, USA.
142. Wang, Z., T. McCarthy, and M. Sheikh. *A comparison of Genetic Algorithm and Particle Swarm Optimisation for Theoretical and Structural Applications*. in *Proceedings of the Eleventh International Conference on*

Computational Structures Technology. 2012. Stirlingshire, UK, Paper 68, doi:10.4203/ccp.99.68: Civil-Comp Press.

APPENDIX A NEW ORCAFLEX INTERFRACE COMPARISON

The new interface files are included on the installation CD from version 9.4 onwards. The use of the interface in Matlab requires the files be added to the Matlab search path in order for Matlab to find them. This can be done via:

```
% add OrcaFlex interface path
addpath 'C:\Program Files\Orcina\OrcaFlex\9.4\MATLAB';
```

If the interface files are not located in the path above, corresponding path should be added into Matlab. The new interface improves the workability in terms of code simplicity and efficiency. Following example shows the function to load an OrcaFlex simulation file and extract the maximum effective tension from range graph for a line using both new and old interface.

```
% New interface example of extracting results
% Create a new model
model = ofxModel;
% load simulation file called 'Test.sim'
model.LoadSimulation('Test.sim');
% extract rangegraph results
line = model('Line1');
rangeResults = line.RangeGraph('Effective Tension');
% display the range graph maximum values
rangeResults.Max
```

```
% Old interface example of extracting results
% Create a new model
loadlibrary OrcFxAPI OrcFxAPI.h
% Define these pointers.
modelh = libpointer('int32Ptr',1);
voidh = libpointer('voidPtr');
status = libpointer('int32Ptr',0);
% Call library function using these pointers. Note: 2
pointers are
% required for output: filename and status.
[filename, status] =
calllib('OrcFxAPI','C_LoadSimulation',...
```

```
    modelh, 'Test.sim', status);
% Generate structure for object handles.
object.ObjectHandle = 0;
object.ObjectType = 0;
object.ObjectName = 0;
% Use libstruct to contruct a structure based on the
structured
% array 'object'.
str_obj = libstruct('TObjectInfo',object);
% Use libpointer to define a pointer for the structured
array
% 'str_obj'.
handles = libpointer('TObjectInfoPtr',str_obj);
[object_name, handles, status] = calllib('OrcFxAPI',...
    'C_ObjectCalled', modelh, handles, 'Line1', status);
% Define variable ID pointer called 'varid'.
varid = libpointer('int32Ptr',0);
% Extract Variable ID for 'Effective Tension'
[variable_ID, varid, status] =
calllib('OrcFxAPI','C_GetVarID',...
    handles.ObjectHandle, 'Effective Tension', varid,
status);
% Get number of range graph points.
[NumRange, status] = calllib('OrcFxAPI',...
    'C_GetRangeGraphNumOfPoints', handles.ObjectHandle,
varid, status);
% Create the required data pointers to extract all Range
Graph
% values. Define a 'ranges' variable to allocate
sufficient memory
% to each of the Range Graph pointers.
ranges(1:NumRange,1) = 0;
Maxval = libpointer('doublePtr',ranges);
% Extract the Range Graph data.
[TPeriod,Maxval,status]
 = calllib('OrcFxAPI', 'C_GetRangeGraph', ...
    handles.ObjectHandle,varid, TPeriod, Maxval, status);
```

From the code comparison example above, it is very prominent that to fulfil the same function in Matlab, the new interface requires much less effort than that of the previous interface. To simply extract effective tension from range graph, new interface required only ten lines of code. In contrast, the previous interface required nearly forty lines of code that was about four times effort than that of new interface. Therefore, using the new interface in the optimisation coding process can improve the working efficiency numerously.

APPENDIX B INTERNSHIP PROJECTS AT AMOG

Riser VIV Fatigue F.O.S Review and Study Proposal

Vortex Induced-Vibration (VIV) based riser fatigue prediction includes the use of a very conservative Fatigue Factor of Safety (FOS) value to guard against the uncertainty of VIV prediction. The existing FOS value for VIV prediction purposes is 20 and stems back to a study performed in 2000. The project was proposed to formulate a VIV FOS for application in riser design that is feasible for different projects.

This was a big project that involved a few phrases. During the placement, the author took the sensitivity analysis of SHEAR7 parameters. Numerous hydrodynamic parameters and coefficients are involved. The effects of uncertainty associated with each hydrodynamic parameters used in SHEAR7 were investigated, including:

- Strouhal number
- Lift coefficients
- Hydrodynamic damping coefficient
- Lock-in range/bandwidth

The sensitivity analysis aimed to find the most important hydrodynamic coefficient governing the SHEAR7 VIV prediction for predicting riser fatigue at a particular location.

FSO Riser System Installation

This project was related to one gas riser and one production riser, which were to be installed into a FSO facility. The author was required to use OrcaFlex to simulate the installation processes. The curvatures of the flexible risers were monitored while

OrcaFlex simulated the lay-down process of the pipes. The speed of the vessel that assures the risers can be laid down without occurring excessive curvatures was recorded.

A deep-water Early Production System (EPS) Study

The concept of EPS was developed to avoid the extreme uncertainties in the deep-water reservoirs that can waste billions of dollars in over-capitalised faculties by investors. At present, there is no such system exist except for the studies and research. The author was part of the team who analysed the bundle riser simulation using OrcaFlex.

Mooring design consultation with experienced engineers

The author had a few conversations with experienced offshore engineers regards to mooring design from a practical engineering perspective during placement, and the details were documented and summarised in the main part of thesis referred to in section 3.9.

APPENDIX C GA & PSO CODES

This section gives the Matlab code that was developed for the GA and PSO for the purpose of benchmark study.

- GA Toolbox codes

```
function [x,fval,exitflag,output,population,score] =
ga_truss(nvars,PopulationSize_Data,EliteCount_Data,StallG
enLimit_Data)
% This is an auto generated M-file from Optimization
Tool.
clear;
clc;
% Start with the default options
options = gaoptimset;
% Modify options setting
options = gaoptimset(options,'PopulationSize', 50);
options = gaoptimset(options,'EliteCount', 2);
options = gaoptimset(options,'Generations',300);
options = gaoptimset(options,'StallGenLimit', 300);
options = gaoptimset(options,'CreationFcn',
@createID_permutations);
options = gaoptimset(options,'SelectionFcn', {
@selectiontournament 2  });
options = gaoptimset(options,'CrossoverFcn',
@crossoverID1_permutation);
options = gaoptimset(options,'MutationFcn',
@mutateID1_permutation);
options = gaoptimset(options,'Display', 'off');
options = gaoptimset(options,'PlotFcns', { @gaplotrange
@trussID_plot });
tic;
[x,fval,exitflag,output,population,score] =
ga(@mass_obj,10,[],[],[],[],[],[],[],options)
toc;
```

- GA creates function

```
function pop =
createID_permutations(NVARS,FitnessFcn,options)
```

```
%CREATEID_PERMUTATIONS Creates a population of
permutations.
%   POP = CREATE_PERMUTATION(NVARS,FITNESSFCN,OPTIONS)
creates a population
%  of permutations POP each with a length of NVARS.
%
%   The arguments to the function are
%     NVARS: Number of variables
%     FITNESSFCN: Fitness function
%     OPTIONS: Options structure used by the GA
totalPopulationSize = sum(options.PopulationSize);
n = NVARS;
m = 42;% 42 IDs could be selected from AISC Table
ini_pop = cell(totalPopulationSize,1);
pop = cell(totalPopulationSize,1);
for i = 1:totalPopulationSize
    ini_pop{i} = randperm(m);
    pop{i} = ini_pop{i}(:,1:n);
end
```

- GA crossover function

```
function xoverKids  =
crossoverID1_permutation(parents,options,NVARS, ...
    FitnessFcn,thisScore,thisPopulation)
%   CROSSOVERID_PERMUTATION Custom crossover function for
10-bar truss.
%
%   The arguments to the function are
%     PARENTS: Parents chosen by the selection function
%     OPTIONS: Options structure created from GAOPTIMSET
%     NVARS: Number of variables
%     FITNESSFCN: Fitness function
%     STATE: State structure used by the GA solver
%     THISSCORE: Vector of scores of the current
population
%     THISPOPULATION: Matrix of individuals in the
current population


nKids = length(parents)/2;
xoverKids = cell(nKids,1); % Normally zeros(nKids,NVARS);
index = 1;
for i=1:nKids
    % here is where the special knowledge that the
population is a cell
    % array is used. Normally, this would be
thisPopulation(parents(index),:);
    parent1 = thisPopulation{parents(index)};
    parent2 = thisPopulation{parents(index+1)};
```

```
    index = index + 2;

    p1 = ceil((length(parent1) -1) * rand);% Random
generate xover point
%     p2 = p1 + ceil((length(parent) - p1- 1) * rand);
    child = [parent1(1:p1) parent2(p1+1:end)];
    xoverKids{i} = child; % Normally, xoverKids(i,:);
end
```

- GA mutates function

```
function mutationChildren = mutateID1_permutation(parents
,options,NVARS, ...
    FitnessFcn, state,
thisScore,thisPopulation,mutationRate)
%   MUTATE_PERMUTATION Custom mutation function for 10-
bar truss.
%
%   The arguments to the function are
%     PARENTS: Parents chosen by the selection function
%     OPTIONS: Options structure created from GAOPTIMSET
%     NVARS: Number of variables
%     FITNESSFCN: Fitness function
%     STATE: State structure used by the GA solver
%     THISSCORE: Vector of scores of the current
population
%     THISPOPULATION: Matrix of individuals in the
current population
%     MUTATIONRATE: Rate of mutation

% Here we swap two elements of the permutation
mutationChildren = cell(length(parents),1);% Normally
zeros(length(parents),NVARS);
for i=1:length(parents)
    parent = thisPopulation{parents(i)}; % Normally
thisPopulation(parents(i),:)
    p = ceil(length(parent) * rand);
    c = ceil(42*rand);
    child = parent;
    child(p)=c;
    mutationChildren{i} = child; % Normally
mutationChildren(i,:)
end
```

- GA analysis codes

```
function [c,ceq] = stressdis_analysis(ID)
%This script calculate stresses of each element
%% Set coordinates for all members
```

```
constant1 = 9.144; % length of element number of 1, 2, 3,
4, 5, 6
constant2 = 12.932; % length of element number of 7, 8,
9, 10
e_modulas = 68.9 * 10^9;
m_density = 2770; %kg/m^3
lstress = [172,172,172,172,172,172,172,172,172,172]; %The
stress limit of each member is 172 MPa
ldisplace = [50.8,50.8,50.8,50.8,50.8,50.8,50.8,50.8];
%The displacement limit of each node in X and Y
direction. Magnify by factor of 172/50.8=3.39
ceq = cell(size(ID,1),1);
ceq(:,1) = {[lstress,ldisplace]};
c = cell(size(ID,1),1);
node1 = [2*constant1,constant1];
node2 = [2*constant1,0];
node3 = [constant1,constant1];
node4 = [constant1,0];
node5 = [0,constant1];
node6 = [0,0];
%%
load('aics.mat');
for count = 1:size(ID,1);
    p = ID{count};
    area1 = t_aics(p(1),2);
    area2 = t_aics(p(2),2);
    area3 = t_aics(p(3),2);
    area4 = t_aics(p(4),2);
    area5 = t_aics(p(5),2);
    area6 = t_aics(p(6),2);
    area7 = t_aics(p(7),2);
    area8 = t_aics(p(8),2);
    area9 = t_aics(p(9),2);
    area10 = t_aics(p(10),2);
    %% Generate global stiffness matrix
    for m=1:10
        if m==1
            gstiff1 = zeros(12,12);
            index = [9,10,5,6];
            m1stiffness = m_stiffness
(area1,e_modulas,constant1,node5(1),node5(2),node3(1),nod
e3(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff1(index(i),index(j))=
m1stiffness(i,j);
                end
            end
        elseif m==2
            gstiff2 = zeros(12,12);
```

```
            index = [5,6,1,2];
            m2stiffness = m_stiffness
(area2,e_modulas,constant1,node3(1),node3(2),node1(1),nod
e1(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff2(index(i),index(j))=
m2stiffness(i,j);
                end
            end
        elseif m==3
            gstiff3 = zeros(12,12);
            index = [11,12,7,8];
            m3stiffness = m_stiffness
(area3,e_modulas,constant1,node6(1),node6(2),node4(1),nod
e4(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff3(index(i),index(j))=
m3stiffness(i,j);
                end
            end
        elseif m==4
            gstiff4 = zeros(12,12);
            index = [7,8,3,4];
            m4stiffness = m_stiffness
(area4,e_modulas,constant1,node4(1),node4(2),node2(1),nod
e2(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff4(index(i),index(j))=
m4stiffness(i,j);
                end
            end
        elseif m==5
            gstiff5 = zeros(12,12);
            index = [5,6,7,8];
            m5stiffness = m_stiffness
(area5,e_modulas,constant1,node3(1),node3(2),node4(1),nod
e4(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff5(index(i),index(j))=
m5stiffness(i,j);
                end
            end
        elseif m==6
            gstiff6 = zeros(12,12);
            index = [1,2,3,4];
```

```
            m6stiffness = m_stiffness
(area6,e_modulas,constant1,node1(1),node1(2),node2(1),nod
e2(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff6(index(i),index(j))=
m6stiffness(i,j);
                end
            end
        elseif m==7
            gstiff7 = zeros(12,12);
            index = [7,8,9,10];
            m7stiffness = m_stiffness
(area7,e_modulas,constant2,node5(1),node5(2),node4(1),nod
e4(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff7(index(i),index(j))=
m7stiffness(i,j);
                end
            end
        elseif m==8
            gstiff8 = zeros(12,12);
            index = [11,12,5,6];
            m8stiffness = m_stiffness
(area8,e_modulas,constant2,node6(1),node6(2),node3(1),nod
e3(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff8(index(i),index(j))=
m8stiffness(i,j);
                end
            end
        elseif m==9
            gstiff9 = zeros(12,12);
            index = [3,4,5,6];
            m9stiffness = m_stiffness
(area9,e_modulas,constant2,node2(1),node2(2),node3(1),nod
e3(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff9(index(i),index(j))=
m9stiffness(i,j);
                end
            end
        elseif m==10
            gstiff10 = zeros(12,12);
            index = [7,8,1,2];
```

```
            m10stiffness = m_stiffness
(area10,e_modulas,constant2,node4(1),node4(2),node1(1),no
de1(2));% call in m_stiffness function
            for i=1:4
                for j=1:4
                    gstiff10(index(i),index(j))=
m10stiffness(i,j);
                end
            end
        end
    end
    gstiff =
gstiff1+gstiff2+gstiff3+gstiff4+gstiff5+gstiff6+gstiff7+g
stiff8+gstiff9+gstiff10;
    %% Create load matrix in Newton
    load1 = [0;0;0;-444822.16;0;0;0;-444822.16];
    %% Create displacement matrix
    displace2 = [0;0;0;0];
    %% Extract submatrix from global stiffness matrix
    submatrix1 = zeros(8,8);
    for k=1:8
        for r=1:8
            submatrix1(k,r) = gstiff(k,r);
        end
    end
    submatrix2 = zeros(4,4);
    for k=9:12
        for r=9:16
            submatrix2(k-8,r-8) = gstiff(k,r-8);
        end
    end
    %% Solve matrix for unkonwns
    displace1 = submatrix1\load1;
    displace3 = displace1';
    load2 = submatrix2 * displace1;
    displace = [displace1;displace2].*10^3; % Total
displacement matrix in mm
    displacement1 = abs(displace3*10^3);% Magnify the
displacement for constraint
    displacement = [displacement1];
    load3 = [load1;load2]./10^3; % Total load matrix in
kN
    for m=1:10
        if m==1
            index = [9,10,5,6];
            m1q = m_load
(area1,e_modulas,constant1,node5(1),node5(2),node3(1),nod
e3(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
```

```
        elseif m==2
            index = [5,6,1,2];
            m2q = m_load
(area2,e_modulas,constant1,node3(1),node3(2),node1(1),nod
e1(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        elseif m==3
            index = [11,12,7,8];
            m3q = m_load
(area3,e_modulas,constant1,node6(1),node6(2),node4(1),nod
e4(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        elseif m==4
            index = [7,8,3,4];
            m4q = m_load
(area4,e_modulas,constant1,node4(1),node4(2),node2(1),nod
e2(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        elseif m==5
            index = [5,6,7,8];
            m5q = m_load
(area5,e_modulas,constant1,node3(1),node3(2),node4(1),nod
e4(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        elseif m==6
            index = [1,2,3,4];
            m6q = m_load
(area6,e_modulas,constant1,node1(1),node1(2),node2(1),nod
e2(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        elseif m==7
            index = [7,8,9,10];
            m7q = m_load
(area7,e_modulas,constant2,node5(1),node5(2),node4(1),nod
e4(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        elseif m==8
            index = [11,12,5,6];
            m8q = m_load
(area8,e_modulas,constant2,node6(1),node6(2),node3(1),nod
e3(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        elseif m==9
```

```
            index = [3,4,5,6];
            m9q = m_load
(area9,e_modulas,constant2,node2(1),node2(2),node3(1),nod
e3(2),displace(index(1)),displace(index(2)),displace(inde
x(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        elseif m==10
            index = [7,8,1,2];
            m10q = m_load
(area10,e_modulas,constant2,node4(1),node4(2),node1(1),no
de1(2),displace(index(1)),displace(index(2)),displace(ind
ex(3)),displace(index(4)))/10^3;% call in m_load function
and conver to kN
        end
    end
    % q = [m1q; m2q; m3q; m4q; m5q; m6q; m7q; m8q; m9q;
m10q];
    %% Get stresses of each member
    [s1,s2,s3,s4,s5,s6,s7,s8,s9,s10] =
m_stress(area1,area2,area3,area4,area5,area6,area7,area8,
area9,area10,m1q, m2q, m3q, m4q, m5q, m6q, m7q, m8q, m9q,
m10q);
    cstress = abs([s1,s2,s3,s4,s5,s6,s7,s8,s9,s10]); %
Stresses in MPa
    c(count,1) = {[cstress,displacement]};
end
```

- PSO codes

```
tic;
np = 200;
dimension = 10;
cost_gbest = 1e50;
cost_lbest = repmat(cost_gbest,np,1);
% random initilize positions of particles
p = createID_permutations(np);
% random initilize velocities
momentum_coeff = 0.875;
c1 = 2;
c2 = 2;
Vel = zeros(np,dimension);
[cost_mass,mass] = mass_obj(p);
ni = 100;
for i = 1:np
    if cost_mass(i) < cost_lbest(i)
        cost_lbest(i) = cost_mass(i);
        lbest(i,:) = p{i}(:);
    end
    if cost_mass(i) < cost_gbest;
        cost_gbest = cost_mass(i);
        gbest(1,:) = p{i}(:);
```

```
    end
end
p_history{1} = p;
mass_history{1} = cost_mass;
penalty_history{1} = mass;
% subplot(2,1,2);
% trussID_plot(gbest)
% subplot(2,1,1);
% score_plot(cost_mass,1)
for i = 2:ni
    for j = 1:np
        Vel(j,:) = Vel(j,:).* momentum_coeff + c1 * rand
.* ...
            (lbest(j,:) - p{j}(1,:)) + c2 * rand * ...
            (gbest - p{j}(1,:));
        temp = Vel>8;
        Vel(temp) = 8;
        temp = find(Vel<-8);
        Vel(temp) = -8;
        p{j,1} = p{j}(1,:) + ceil(Vel(j,:));
        % check compliance
        for k = 1:dimension
            if p{j}(1,k) < 1
%                 % violated design points redirection
%                 Vel(j,k) = c1 * rand .* ...
%                     (lbest(j,k) - p{j}(1,k)) + c2 *
rand * ...
%                     (gbest(k) - p{j}(1,k));
%                 p{j}(1,k) = p{j}(1,k) + ceil(Vel(j,k));
%                 if p{j}(1,k) < 1
                  p{j}(1,k) = 2;
%                 end
            elseif p{j}(1,k) > 42
%                 Vel(j,k) = c1 * rand .* ...
%                     (lbest(j,k) - p{j}(1,k)) + c2 *
rand * ...
%                     (gbest(k) - p{j}(1,k));
%                 p{j}(1,k) = p{j}(1,k) + ceil(Vel(j,k));
%                 if p{j}(1,k) > 42
                  p{j}(1,k) = 41;
%                 end
            end
        end
    end
    [cost_mass,mass] = mass_obj(p);
    for j = 1:np
        if cost_mass(j) < cost_lbest(j)
            cost_lbest(j) = cost_mass(j);
            lbest(j,:) = p{j}(:);
        end
```

```
        if cost_mass(j) < cost_gbest;
            cost_gbest = cost_mass(j);
            gbest(1,:) = p{j}(:);
            fprintf('The gbest appears in %2.2f
iteration.\n',i)
        end
    end
%     subplot(2,1,2);
%     trussID_plot(gbest)
%     subplot(2,1,1);
%     swarm_plot(p,cost_mass,np,i);
%     score_plot(cost_mass,i)
%     drawnow
    p_history{i} = p;
    mass_history{i} = cost_mass;
    penalty_history{i} = mass;
end
toc;
% x{1} = gbest;
% [totalmass,mass] = mass_obj(x)
% x{1}
%%
```

- PSO mass function

```
function [totalmass,mass,massps,masspd] = mass_obj(ID)
% This is the fitness function which calculate the mass
of the whole
% structure
%% Set cordinates for all members
constant1 = 9.144; % length of element number of 1, 2, 3,
4, 5, 6
constant2 = 12.932; % length of element number of 7, 8,
9, 10
% e_modulas = 68.9 * 10^9;
m_density = 2770; %kg/m^3
mass = zeros(size(ID,1),1);
massps = zeros(size(ID,1),1);
masspd = zeros(size(ID,1),1);
totalmass = zeros(size(ID,1),1);
%% Calculate stresses and displacement
stressdis = stressdis_analysis(ID);% Truss analysis


%% Get mass & penalty mass of each member
for i = 1:size(ID,1);
    p = ID{i};
    mass(i) =
f_fitness(constant1,constant2,m_density,p(1),p(2),p(3),p(
4),p(5),p(6),p(7),p(8),p(9),p(10));
```

```
    massps(i) = stress_penalty(stressdis{i}(1:10));
    masspd(i) = dis_penalty(stressdis{i}(11:18));
    totalmass(i) = mass(i)+ massps(i)+ masspd(i);
end
```

APPENDIX D RAMNAS TECHNICAL BROCHURE

Proof and break loads (kN)

LOAD (in kN) = c x d^2 x (44 – 0.08 x d) (d in mm)

Test Load	Break Load						Proof Load											Weight	
Grade	ORQ	R3/ NVR3	R3S	R4	R4S	R5	ORQ	R3	NVR3	R3S Stud	R3S Studless	R4 Stud	R4 Studless	R4S Stud	R4S Studless	R5 Stud	R5 Studless		
C-factor	0,0211	0,0223	0,0249	0,0274	0,0304	0,032	0,014	0,0148	0,0156	0,018	0,0174	0,0216	0,0192	0,024	0,0213	0,0251	0,0223	Stud	Studless
mm																		kg/m	
70	3970	4196	4685	5156	5720	6021	2634	2785	2935	3387	3274	4064	3613	4516	4008	4723	4196	107	98
73	4291	4535	5064	5572	6182	6507	2847	3010	3172	3660	3538	4392	3904	4881	4331	5104	4535	117	107
76	4621	4884	5454	6001	6658	7009	3066	3242	3417	3942	3811	4731	4205	5257	4665	5498	4884	126	116
78	4847	5123	5720	6295	6984	7351	3216	3400	3584	4135	3997	4962	4411	5514	4893	5766	5123	133	122
81	5194	5490	6130	6745	7484	7877	3446	3643	3840	4431	4283	5317	4726	5908	5243	6179	5490	144	131
84	5550	5866	6550	7208	7997	8418	3683	3893	4104	4735	4577	5682	5051	6313	5603	6602	5866	155	141
87	5916	6252	6981	7682	8523	8971	3925	4149	4374	5046	4878	6056	5383	6729	5972	7037	6252	166	151
90	6289	6647	7422	8167	9062	9539	4173	4412	4650	5365	5187	6439	5723	7154	6349	7482	6647	177	162
92	6544	6916	7722	8497	9428	9924	4342	4590	4838	5582	5396	6699	5954	7443	6606	7784	6916	185	169
95	6932	7326	8180	9001	9987	10512	4599	4862	5125	5913	5716	7096	6307	7884	6997	8246	7326	198	181
97	7195	7604	8490	9343	10366	10911	4774	5047	5319	6138	5933	7365	6547	8184	7263	8559	7604	206	188
100	7596	8028	8964	9864	10944	11520	5040	5328	5616	6480	6264	7776	6912	8640	7668	9036	8028	219	200
102	7868	8315	9285	10217	11336	11932	5220	5519	5817	6712	6488	8054	7159	8949	7942	9359	8315	228	208
105	8282	8753	9773	10754	11932	12560	5495	5809	6123	7065	6829	8478	7536	9420	8360	9851	8753	241	221
107	8561	9048	10103	11118	12335	12984	5681	6005	6330	7304	7060	8764	7790	9738	8643	10184	9048	251	229
111	9130	9650	10775	11856	13154	13847	6058	6404	6750	7789	7529	9347	8308	10385	9217	10861	9650	270	246
114	9565	10109	11287	12420	13780	14506	6346	6709	7071	8159	7887	9791	8703	10879	9655	11378	10109	285	260
117	10005	10574	11807	12993	14415	15174	6639	7018	7397	8535	8251	10242	9104	11380	10100	11902	10574	300	274
120	10452	11047	12334	13573	15059	15852	6935	7331	7728	8916	8619	10700	9511	11889	10551	12434	11047	315	288
122	10753	11365	12690	13964	15493	16308	7135	7542	7950	9173	8868	11008	9785	12231	10855	12792	11365	326	298
124	11057	11686	13048	14358	15930	16768	7336	7755	8175	9432	9118	11319	10061	12576	11161	13153	11686	337	308
127	11516	12171	13591	14955	16592	17466	7641	8078	8515	9824	9497	11789	10479	13099	11626	13700	12171	353	323
130	11981	12663	14139	15559	17262	18171	7950	8404	8858	10221	9880	12265	10903	13628	12095	14253	12663	370	338
132	12294	12993	14508	15965	17713	18645	8157	8623	9089	10488	10138	12585	11187	13984	12411	14625	12993	382	348
137	13085	13829	15441	16992	18852	19844	8682	9178	9674	11162	10790	13395	11906	14883	13209	15565	13829	411	375
142	13887	14677	16388	18033	20008	21061	9214	9741	10267	11847	11452	14216	12637	15796	14019	16520	14677	442	403
147	14700	15536	17347	19089	21179	22294	9753	10311	10868	12540	12122	15048	13376	16720	14839	17487	15536	473	432
152	15522	16405	18317	20156	22363	23540	10299	10887	11476	13241	12800	15890	14124	17655	15669	18464	16405	506	462
157	16352	17282	19297	21234	23559	24799	10850	11469	12089	13949	13484	16739	14879	18599	16507	19452	17282	540	493
162	17188	18166	20284	22320	24764	26068	11405	12056	12708	14663	14174	17596	15641	19551	17351	20447	18166	575	525
167	18030	19056	21278	23414	25977	27345	11963	12647	13330	15381	14869	18458	16407	20508	18201	21448	19056	611	558
172	18876	19950	22276	24513	27196	28628	12525	13240	13956	16103	15566	19324	17177	21471	19055	22455	19950	648	592
177	19725	20847	23278	25615	28420	29915	13088	13836	14584	16827	16267	20193	17949	22437	19912	23465	20847	686	627

Due to the application of different rounding-off-principles for calculation of loads, individual classification societies show slightly different load values in their tables.
Conversion factors, see page 9.