Lund University GEM thesis series nr 11

Multi-Objective Optimisation Algorithms for GIS-based Multi-Criteria Decision Analysis: An application for evacuation planning

José Ignacio Díaz González

2016
Department of
Physical Geography and Ecosystem Science
Lund University
Sölvegatan 12
S-223 62 Lund
Sweden

José Ignacio Díaz González (2016)
Multi-Objective Optimisation Algorithms for GIS-based Multi-Criteria Decision Analysis: An application for evacuation planning
Master degree thesis, 30 credits in Geo-information Science and Earth Observation for Environmental Modelling and Management
Department of Physical Geography and Ecosystem Science, Lund University

Level: Master of Science (MSc)

Course duration: September 2014 until June 2016

Disclaimer

Consortium partners

The GEM master program is a cooperation of departments at five different universities:

Lund University (Sweden)
University of Warsaw (Poland)
Southampton University (United Kingdom)
University of Twente, ITC (The Netherlands)
University of Iceland (Iceland)

Multi-Objective Optimisation Algorithms for GIS-based Multi-Criteria Decision Analysis: An application for evacuation planning

---

José Ignacio Díaz González

Master thesis, 30 credits, in Geo-information Science and Earth Observation for Environmental Modelling and Management

Supervisor: Ali Mansourian
Lund University

Examiner: Lars Harrie
Lund University

Committee: Petter Pilesjö
Lund University

01001000 01101111 01100100 01101111 01110010
00101110 00101110 00101110

Abstract

Geographic Information Systems have gradually acquired greater relevance as tools to support decision-making processes, and during the last decades they have been used in conjunction with Multi-Criteria Decision Analysis techniques (GIS-MCDA) to solve real-world spatial problems.

GIS-MCDA can be generally divided in two main approaches: Multi-Attribute and Multi-Objective techniques. Until now most of the applications of GIS-MCDA have been focused only on using the multi-attribute approach, and less than 10% of the research has been related to a specific type of multi-objective technique: the use of heuristic/meta-heuristic algorithms.

The present study explores how different heuristic/meta-heuristic methods perform on solving a spatial multi-objective optimisation problem. To achieve this, four algorithms representing different types of heuristics methods were implemented, and applied to solve the same spatial multi-objective optimisation problem related with an evacuation planning situation. The implemented algorithms were Standard Particle Swarm Optimisation (SPSO), Non-dominated Sorting Genetic Algorithm II (NSGA-II), Archived Multi-Objective Simulated Annealing (AMOSA) and Multi-Objective Grey Wolf Optimiser (MOGWO).

The results show that the four algorithms were effective on solving the given problem, although in general AMOSA and MOGWO had a higher performance for the evaluated aspects (number of solutions, effectiveness of the optimisation, diversity, execution time and repeatability). However, the differences in the results obtained from each algorithm were not clear enough to state that one type of heuristic is superior than others. Since AMOSA and MOGWO are the most recent algorithms among the implemented ones, they include several improvements achieved by the latest research, and their superior performance could be connected to these improvements more than to the specific type of algorithms they belong to.

Further research is suggested to explore the suitability of these methods for many-objectives spatial problems, to consider the temporal variability and dynamism of real-world situations, to create a standard set of algorithms to be used for benchmarking, and to integrate them with the currently available GIS-MCDA tools. Despite this, from the performed research it is possible to conclude that heuristics methods are reliable techniques for solving spatial problems with multiple and conflictive objectives, and future research and practical implementations in this field can strengthen the capacities of GIS as a multi-criteria decision-making support tool.

KEYWORDS: GIS, multi-criteria, multi-objective, optimisation, decision, analysis, decision-making, algorithm, SPSO, NSGA-II, AMOSA, MOGWO.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Background

Geographic Information Systems (GIS) have gradually acquired greater relevance as tools to support decision-making processes. Their capacities to store, manipulate and analyse spatial data have provided valuable information for decision-making in diverse areas, such as natural resources management, hazard control, regional and urban planning (Jankowski 1995). From one perspective, the final aim of GIS is to provide support for making decisions, not only for experts in geographical and environmental sciences, but also to decision-makers and stakeholders from different domains and interests (Jankowski and Nyerges 2003; Malczewski and Rinner 2015).

For its part, Multi-Criteria Decision Analysis (MCDA) is a branch of the Operational Research domain which seeks to support decision-making when problems consider multiple variables, which often could be conflicting with each other (Carver 1991; Malczewski 2006). Normally, in these cases there is no a single solution that can be identified as "the best", but there is a set of "good" alternatives which meet the defined objectives in different proportions. Thus, the general objective of MCDA is to assist the decision-makers in selecting the "best" alternative from that set of good solutions when there are multiple choice criteria and priorities (Jankowski 1995).

Although usually we do not notice it, multi-criteria analyses are part of our daily lives. The purchase of goods and services is a simple example of this, where we want to get the highest possible benefit paying the lowest price (two objectives). However, likely the best TV set or the best vacation plan will not be the cheapest options, and there will be a range of price-quality combinations to choose from (the set of good solutions). Then, the final decision will be done by applying one or more personal preference (the constraints), such as "*I am willing to pay a little more to buy this model with a better design*" or "*This is the best product I can buy with the money I am willing to spend*".

In the same way, real-world problems with spatial characteristics also have to deal with these multiple criteria and potentially conflicting objectives. For instance, the promoters of a water

dam want to maximise the storage capacity but, at the same time, minimise construction cost and water loss due to evaporation (or infiltration). The set of possible dam designs (alternatives) allows many different choices. The criteria are functions of the decision variables to be maximised or minimised, and they are clearly in conflict: a dam with big storage capacity will certainly not involve small construction cost (Ehrgott 2005), or a bigger capacity of the lake will imply a higher loss of water.

Analysing the optimal routes to a certain destination or assessing the environmental impact of an activity are also examples of spatial problems involving multiple decision criteria (Huang et al. 2011). In these real-world situations there is no a single “best” solution that accomplishes all the objectives in the best way, and it is for trying to solve this type of problems that GIS and MCDA intersect their domains, generating a positive synergy as tools to support decision-making (Malczewski and Rinner 2015).

Since the decade of 1980 several MCDA techniques have been applied to GIS (GIS-MCDA) although the way to classify them widely varies in the literature depending on the point of view. Because of this variety of classification (often overlapped among them), for this study the techniques of GIS-MCDA have been grouped into two main types of methods: Multi-Attribute Decision Analysis (MADA) and Multi-Objective Decision Analysis (MODA) (Ehrgott et al. 2010).

In general terms, MADA methods are those involving a limited number of known attributes (for instance, vegetation cover, slope or population density). Then, the solutions are found applying several constraints and assessing the relative relevance of each attribute. By doing this an overall "quality" value for the solution is obtained, which can be compared to the quality of other solutions to finally select the best ones.

MADA methods are the most widely implemented type of GIS-MCDA so far and nowadays there is a wide variety of available tools to implement them, both proprietary and free/open source software (University of Redlands and SDS Consortium 2009). The Weighted Linear Combination approach (WLC), also called “Weighted summation”, “Weighted linear average”, “Simple additive weighting” or “Weighted overlay” is the best example of this type of techniques (Malczewski 2006).

MODA methods are those techniques that use a set of objective functions (also called cost functions) and a set of constraints defined for each decision variable, in order to evaluate the quality of the candidate solutions (also called "the fitness") (Coello et al. 2007). In these methods the set of good solutions is generated following the principles of Pareto optimality theory (Ehrgott 2005).

Once again, the MODA methods can be subdivided according to their way of solving problems. The first subgroup corresponds to the deterministic methods and are based on the principles of mathematical programming. Most of the current implementations of MODA in

GIS-MCDA correspond to this subgroup, which have in common that the multi-objective problem is transformed into a scalar function, and then it is solved as a single-objective optimisation problem (Malczewski 2006). However, this deterministic approach is not efficient in solving complex spatial multi-objective problems, and in some cases it is not even possible to search for every candidate solution using the classic mathematical programming methods (Carver 1991; Malczewski and Rinner 2015).

The second subgroup of MODA methods corresponds to the stochastic ones, which consider the use of algorithms to overcome the intrinsic limitations of deterministic methods (Malczewski 2006; Coello et al. 2007; Coello 2009; Ehrgott et al. 2010). Using a trial and error approach, these algorithms utilise several techniques to refine each solution and to optimise the objective functions (minimising or maximising them). They also incorporate different levels of randomness to ensure a proper exploration of the search space and to avoid local maxima or minima (Deb et al. 2002; Bandyopadhyay et al. 2008; Bonyadi and Michalewicz 2014; Mirjalili et al. 2014).

1.2 Problem statement

According to the literature, the subgroup of stochastic MODA methods is so far the less studied branch of GIS-MCDA. Although since the decade of 2000 the research on this subject has been increased, there is still a lack of studies about using heuristics/meta-heuristics to solve spatial optimisation problems and the available studies represent less than 10% of the total research on GIS-MCDA (Ehrgott et al. 2010; Malczewski and Rinner 2015). Moreover, most of this research has been focused only in a few type of heuristics, with the “evolutionary algorithms” being the most studied type until now.

This lack of studies would be, indeed, one of the main challenges to face by GIS-MCDA researchers, in order to convince sceptical decision-makers about the feasibility of this approach to produce trustworthy results (Zheng et al. 2015). If stochastic methods are properly defined and implemented they can decrease computational times and solve more complex spatial multi-objective optimisation problems, generating more interactive decision-making contexts (Church et al. 2003 and Duh and Brown 2005, cited in Malczewski and Rinner 2015).

The general approach used in previous MODA studies has considered the use of one heuristic/meta-heuristic algorithm to solve a given spatial problem and then the analysis of its performance (Ngamchai and Lovell 2003; Saadatseresht et al. 2009; Sasaki et al. 2010; Demetriou et al. 2014; Shaygan et al. 2014; Son 2014). However, there has not been extensive research on analysing if some types of algorithms are better than others to solve spatial problems.

Therefore, there is a gap in the existent research about using heuristic/meta-heuristic algorithms to solve spatial multi-objective optimisation problems, and researching in this field is an opportunity to strengthen the role of GIS as a decision-making support tool.

1.3 Objectives

Based on the previous studies in this field and the identified gap, the objectives of this research are:

- To explore how four different types of heuristic/meta-heuristic methods perform in solving a spatial multi-objective optimisation problem.

- To analyse if the quality of the obtained solutions depends on the type of algorithm being used and if a specific type of heuristic is more suitable than others to solve the given problem.

- To compare the optimised solutions against the solutions for a base case, in which no multi-objective optimisation is applied.

To achieve this, several heuristics algorithms have been implemented and applied to solve the same spatial multi-objective optimisation problem, which is related with a real-world situation of evacuation planning.

1.4 Outline

This document is organised in six chapters, being Chapter 1 this Introduction. Chapter 2 presents the theoretical background of the research, introducing some general aspects of the MCDA domain and more specific details about the state-of-the-art in the GIS-MCDA research field. Chapter 3 describes the methods and the heuristics that were implemented, as well as the evacuation planning optimisation problem used as a case study. Chapter 4 presents the results and solutions obtained using each method. In Chapter 5 the results are analysed and discussed regarding the declared aims of this research. Chapter 6 summarises the conclusions yielded from the previous analysis, and finally a list of bibliographical references used to support this research and one Appendix with complementary data are also presented.

Chapter 2
Theoretical background

2.1 Definitions

In order to develop an understanding of Multi-Objective Optimisation Problems (MOP or MOPs) a series of formal non-ambiguous definitions are required (Coello et al. 2007). Several definitions can be found in the literature, some of them slightly different but conceptually equivalent. The following list of definitions does not intend to be an extensive vocabulary but just to introduce the reader to some basic terminology frequently used in MCDA, adjusted to the context of this research.

- *Multi-Objective optimisation problem*: in words, a MOP can be defined as the problem of finding a vector of decision variables which satisfies constraints and optimises a vector function, whose elements represent the objective functions. These functions form a mathematical description of performance criteria, which are usually in conflict with each other. Hence, the term "optimise" means finding such a solution which would give the values of all the objective functions acceptable to the decision maker (Coello et al. 2007). This statement can be mathematically expressed as follows:

$$\text{minimise (or maximise)} \qquad F(x)=\{f_1(x), f_2(x), ..., f_n(x)\} \qquad (2.1)$$

$$\text{subject to:} \quad x \in X$$

  where $F(x)$ is the n-dimensional objective function, $f_k(x)$ is an objective function (k = 1, 2, ..., n), X is the set of feasible alternatives and $x = (x_1, x_2, ..., x_m)$ is a vector of decision variables, $x_i \geq 0$, for i = 1, 2, ..., m (Coello et al. 2007; Malczewski and Rinner 2015).

- *Decision variables*: are the numerical quantities for which values are to be chosen in an optimisation problem. These quantities are denoted as x_j, where j = 1, 2, …., n (Coello et al. 2007). For example, a spatial problem like location/allocation can be used for defining a set of spatial options. The locational alternatives could be defined as binary vectors, $x = (x_1, x_2, ..., x_m)$, where a decision variable x_j receives a value of

"1" if an activity is located at the *i*th site (health service, supermarkets, etc.) or a value of "0" otherwise. Also, a vector of allocation variables associated with the *j*th location can be defined in terms of a binary variable x_{ij} receiving a value of "1" if an activity (demand for health services) at the *i*th location is allocated to the *j*th location, or a value "0" otherwise (Malczewski and Rinner 2015).

- *Solution*: in a broad sense it means any well-formed answer to the problem that maps to a cost through the function *f*. The objective is not to find a solution, but to find a minimum cost one. Therefore, it is meaningful to talk about an approximate solution to the problem, i.e., one that is close to optimal (Knowles et al. 2008).

- *Constraints*: in most optimisation problems there are always restrictions imposed by the particular characteristics of the environment or available resources (for instance, physical limitations or time restrictions). These restrictions must be satisfied in order to consider a certain solution acceptable. All these restrictions in general are called constraints, and they describe dependences among decision variables and constants (or parameters) involved in the problem (Coello et al. 2007).

- *Objective function and fitness*: The objective function is the statement of the goal of an optimisation problem (Kennedy et al. 2001), the mathematical expression to be evaluated. The fitness is a measure of the degree to which a candidate solution successfully solves the problem being addressed (Kennedy et al. 2001). In MOPs, it is a vector obtained after evaluating the solution for all the objectives functions.

- *Search space*: also called "objective space" or "objective function space" is used to denote the coordinate space within which vectors resulting from evaluating an MOP's solutions are plotted (Coello et al. 2007).

- *Pareto Optimality*: having several objective functions the notion of "optimum" is different to a single-objective problem, because in MOPs the aim is to find good compromises (or "trade-offs") rather than a single solution as in global optimisation. The notion of "optimum" most commonly adopted was generalised by Vilfredo Pareto (Coello et al. 2007), which, in words, says that x^* is Pareto optimal if there exists no feasible vector x which would decrease some criterion without causing a simultaneous increase in at least one other criterion (assuming minimisation)(Coello et al. 2007).

- *Dominance*: Since in a MOP the quality of a candidate solution is now no longer measured as a scalar but as a vector, a different way to assess whether or not some solution x is better than a solution y is used: i) It is said that "x and y are equally fit" if their decision vectors (their fitnesses) are identical; ii) It is said that "x is better than y" if x's fitness is better than y's in at least one objective and no worse in all the others. This is called dominance and in such case it is said that "x dominates y"; iii) It is possible a case in which x is better than y on some objectives, but y is better than x on

other objectives (none of them is better than the other in all the aspects). In this situation it is said that "*x* and *y* are non-dominated solutions" (Knowles et al. 2008).

- *Pareto set and Pareto front*: given a set of multi-objective solutions, some of them will be dominated by others in the set. Those that are not dominated by any others in that set (which may be a single solution, or the whole set) form what is called the Pareto set (Knowles et al. 2008). When plotted in the objective space, the Pareto set is collectively known as the Pareto front (Coello et al. 2007).

2.2 Methods classification

As mentioned previously, several schema exist to classify techniques and methods applied in GIS-MCDA. Most of these procedures have been taken from the general decision theory (Malczewski 1999) and therefore, some of those schema have also been ported from this theory. One way of organising them, which is the way to be applied in this report, is taking into account the criteria used during the decision processes to search for the solutions, which can be Attributes (MADA) or Objectives (MODA) (Hwang and Yoon 1981, cited in Ehrgott et al. 2010; Malczewski and Rinner 2015) as shown in Table 2.1.

This research is specifically focused in the stochastic subset of the MODA methods, although next section briefly presents a summary of the relevant studies using both MADA and MODA approaches, in order to give an overview about the GIS-MCDA practices so far which can be later extended by reading specific literature.

Table 2.1: Multi-Attribute and multi-objective decision analysis approaches

Condition	**Multi-Attribute Decision Analysis (MADA)**	**Multi-Objective Decision Analysis (MODA)**
Criteria defined by	Attributes	Objectives
Objectives defined	Implicitly	Explicitly
Attributes defined	Explicitly	Implicitly
Constrains defined	Implicitly	Explicitly
Alternatives defined	Explicitly	Implicitly
Decision modelling paradigm	Outcome-oriented evaluation/choice	Process-oriented design/search

Source: Modified from Hwang and Yoon 1981 and Malczewski 1999, cited in Malczewski and Rinner 2015.

2.3 Previous research in GIS-MCDA

Malczewski (2006) performed an extensive literature review on GIS-MCDA initially including more than 300 articles published in refereed journals between 1990 and 2004. The list was later expanded by Malczewski and Rinner (2015) in order to include articles

published between 2005 and 2010 yielding a total of 805 articles, which progression in time is showed in Figure 2.1.

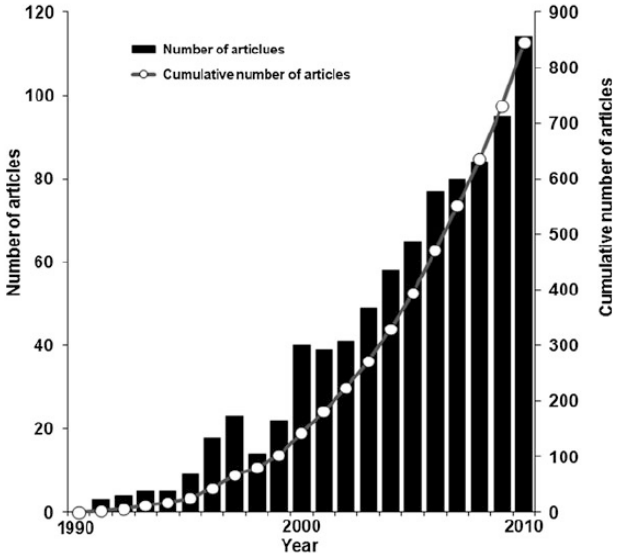


Figure 2.1: GIS-MCDA articles published between 1990 and 2010 (from Malczewski 2006, updated by Malczewski and Rinner 2015).

As can be seen in the figure, the accumulated research on GIS-MCDA has constantly grown since the beginning of the 1990 decade, when it was almost null. The number of articles has also increased almost every year of the period, being the period since 2005 the most productive in terms of publications, accumulating around of 70% of the total.

According to Malczewski and Rinner (2015) this increase can be explained by two main reasons: i) The increase in the computational capacities of personal computers as well as the refinement, development and implementation of more sophisticated and user-oriented GIS and decision analysis software; ii) A general recognition within the GIS community about the relevance of GIS as a tool to support decision analysis.

However, in terms of approaches the studies have been unequally distributed so far. Ehrgott et al. (2010) presented a summary of the published literature differentiating between MADA and MODA methods (based on Malczewski's original survey) which showed that around 71% of the total research belonged to the MADA approach and only about 29% was done about the MODA type. Moreover, only 8% of the total was about heuristic/meta-heuristic methods (which is the main subject of this research), as is shown in Table 2.2.

Table 2.2: Percentage of GIS-MCDA research by type of method

Type	Decision rule	%
MADA	Weighted summation/overlay	39.4
	Ideal/reference point	9.6
	Analytical Hierarchy Process (AHP)	9.4
	Outranking methods (ELECTRE, PROMETHEE)	4.7
	Other	8.3
	Subtotal	**71.3**
MODA	Multi-Objective programming algorithms (linear-integer programming)	15.7
	Heuristic search/evolutionary/genetic algorithms	8.0
	Goal programming/reference point algorithms	2.5
	Other	2.5
	Subtotal	**28.7**
	Total	100.0

Source: Adapted from Ehrgott et al. 2010.

2.3.1 Main aspects of MADA methods

Despite that a large number of multi-attribute decision-making methods have been described in the literature, from Table 2.2 it is clear that just a few of them have actually been implemented or studied in a GIS context. According to Malczewski and Rinner (2015) the Weighted Linear Combination (WLC) model and related methods are the most widely used techniques, not only for MADA but in general for GIS-MCDA. Even more, if we consider that Analytical Hierarchy Process methods are a particular case of WLC, almost 50% of the total research in GIS-MCDA has been performed only for this class of multi-criteria analysis technique. Because of this, most of the so-called "*GIS-based Multi-Criteria analysis tools*" currently available are actually using this specific approach.

In simple terms, WLC-related techniques are composed by a set of criterion weights (w_k) and value functions ($v(a_{ik})$). Then each *i*th decision alternative is associated with a set of criterion weights and subsequently combined with the attribute values $a_{i1}, a_{i2}, \dots, a_{in}$ (with $i = 1, 2, \dots, m$)(Malczewski and Rinner 2015). The mathematical expression to summarise this method would be in the following form:

$$V(A_i) = \sum_{k=1}^{n} w_k v(a_{ik}) \qquad (2.2)$$

where in spatial terms $V(A_i)$ would be the overall value of the *i*th alternative at a certain location *i*, and $v(a_{ik})$ would be the value of the *i*th alternative with respect to the *k*th attribute (evaluated using the value functions). The alternative with the highest value of $V(A_i)$ would be the best one among the assessed options (Malczewski and Rinner 2015).

The main advantage of WLC and the reason behind its extensive use is that WLC-related techniques can easily be implemented in GIS environments, just by using map algebra operations and cartographic modelling (Tomlin 1990 cited in Malczewski 2000). This method is also easy-to-understand to decision makers (Hwang and Yoon 1981, Massam 1988, cited in Malczewski 2000).

Several software alternatives performing MADA and WLC-related analyses are available nowadays in the market, including both commercial and open/free licenses. Most of modern GIS desktop software (for instance Quantum GIS or ArcGIS) have native capacities to work with these methods. There are also several add-ons that extend these inherit capacities and other GIS software, like IDRISI or ILWIS, have also developed specific modules to handle these types of analyses. On the other hand projects like DSMCE (Boerboom 2012) and similar have ported the MADA methods and the WLC model to the web, facilitating the general access to it.

Further details about available software options to support decision-making using these techniques can be found in the literature (Jankowski 1995; Malczewski 1999; Jankowski and Nyerges 2003; Malczewski 2006; Greene et al. 2011; Malczewski and Rinner 2015) or in the web (University of Redlands and SDS Consortium 2009).

2.3.2 Main aspects of MODA methods

From Table 2.2 it is possible to observe that over 50% of the research about MODA corresponds to linear-programming related methods, which combine the objectives of a multi-objective problem with a set of weights provided by decision-makers (that defines the relative importance of each objective). Doing so a single-objective model is created and then a conventional mathematical programming algorithm can be used to solve the problem (Mirjalili et al. 2016).

As can be inferred, using the previous approach requires to know in advance the set of weights provided by the decision-makers. Also, if the optimisation process delivers only a single solution it will be necessary to repeat it as many times as solutions are needed, changing the weights configurations before each run. The use of weights could also introduce certain degree of bias in the final set of solutions, since they are defined based on the expertise or particular interests of the decision-makers.

On the other hand, by using stochastic algorithms the multi-objective formulation of the original problem is kept, allowing to explore the behaviour of the problems across the whole range of design parameters and operating conditions. By doing this the output of the algorithm will be a set of traded-off solutions, from which the decision-makers will eventually choose one according to their needs (Mirjalili et al. 2016).

In general there are two types of stochastic algorithms: heuristic and meta-heuristic, though their difference is small. Heuristic means “to find” or “to discover by trial and error”. Quality solutions can be found in a reasonable amount of time, but there is no guarantee that optimal solutions will be reached. This is good when the best solutions are not needed, but rather good solutions which are easily reachable (Yang 2010). A further development of the heuristic algorithms are the so-called meta-heuristic ones (where “meta” means “beyond” or “higher level”), and they generally perform better than simple heuristics using certain trade-off of randomization and local search (Yang 2010).

Although the recent trend tends to name all stochastic algorithms with randomisation and local search as meta-heuristic, no agreed definitions exist in the literature and both terms are often used interchangeably (Yang 2010). This interchangeable use between heuristic and meta-heuristic terms is also used in this research.

Even if the heuristic methods do not guarantee that an optimal solution will be ever found, its approach along with several techniques to iteratively improve the candidate solutions give them good chances of finding a set of good-enough alternatives (Ehrgott 2005; Coello et al. 2007; Malczewski and Rinner 2015). Additionally, since all the objectives are equally treated during the optimisation process (no preference) the decisions are taken assessing an actual set of traded-off alternatives where no previous bias was applied.

In addition, since a set of potential solutions can be obtained after a single run of the algorithms, these options can be useful for solving complex or large spatial problems where the linear-programming approaches are not efficient, for example, because of its computational complexity (Ehrgott et al. 2010; Malczewski and Rinner 2015).

2.3.3 Choosing between MADA and MODA methods

There is no a rule of thumb that can be applied to decide about the most adequate method to handle certain spatial problem, and because of the high amount of available methods the selection depends much on the context. However, some clues can help on the selection of the most suitable method for a given problem.

As a first filter, a separation can be done whether or not there are multiple objectives. If the decision-makers determine that the multiple objectives are either complementary or can be prioritised, then MADA methods can be applied for solving it (Greene et al. 2011).

On the contrary, if the multiple objectives are in conflict MODA methods are therefore required, and the choice could be based on the complexity of the problem or the desired output (mathematical programming for locating an optimal solution or heuristic methods for locating a set of satisfactory solutions close to the optimum) (Greene et al. 2011).

Unfortunately the available literature does not provide a definition of what could be considered as a complex or large problem, as it also depends on the computational capacity of the software or algorithms being used. However, these methods should not be considered as mutually exclusive, and could be separately applied to different stages of the same decision process (Greene et al. 2011). Multiple techniques can also be applied in parallel as part of a strategy to validate the robustness of the recommendations (Carver 1991 and Roy 2005, cited in Greene et al. 2011).

2.4 Nature-inspired algorithms

2.4.1 Overview

Nature-inspired algorithms try to solve a given problem mimicking the behaviour of natural species or the rules of natural phenomena. Although not all of them are efficient, a few algorithms have proved their capacities for solving real-world problems. Thus, they have become popular within the communities of optimisation, computational intelligence and computer sciences, and they are now among the most widely used algorithms for optimisation and computational intelligence (Yang 2014). For example, Fister Jr. et al. (2013) performed a review of nature-inspired algorithms for solving optimisation problems, finding more than 70 alternatives shown in the Figure 2.2.

Swarm intelligence based algorithms			Bio-inspired (not SI-based) algorithms		
Algorithm	Author	Reference	Algorithm	Author	Reference
Accelerated PSO	Yang et al.	[69], [71]	Atmosphere clouds model	Yan and Hao	[67]
Ant colony optimization	Dorigo	[15]	Biogeography-based optimization	Simon	[56]
Artificial bee colony	Karaboga and Basturk	[31]	Brain Storm Optimization	Shi	[55]
Bacterial foraging	Passino	[46]	Differential evolution	Storn and Price	[57]
Bacterial-GA Foraging	Chen et al.	[6]	Dolphin echolocation	Kaveh and Farhoudi	[33]
Bat algorithm	Yang	[78]	Japanese tree frogs calling	Hernández and Blum	[28]
Bee colony optimization	Teodorović and Dell'Orco	[62]	Eco-inspired evolutionary algorithm	Parpinelli and Lopes	[45]
Bee system	Lucic and Teodorovic	[40]	Egyptian Vulture	Sur et al.	[59]
BeeHive	Wedde et al.	[65]	Fish-school Search	Lima et al.	[14], [3]
Wolf search	Tang et al.	[61]	Flower pollination algorithm	Yang	[72], [76]
Bees algorithms	Pham et al.	[47]	Gene expression	Ferreira	[19]
Bees swarm optimization	Drias et al.	[16]	Great salmon run	Mozaffari	[43]
Bumblebees	Comellas and Martinez	[12]	Group search optimizer	He et al.	[26]
Cat swarm	Chu et al.	[7]	Human-Inspired Algorithm	Zhang et al.	[80]
Consultant-guided search	Iordache	[29]	Invasive weed optimization	Mehrabian and Lucas	[42]
Cuckoo search	Yang and Deb	[74]	Marriage in honey bees	Abbass	[1]
Eagle strategy	Yang and Deb	[75]	OptBees	Maia et al.	[41]
Fast bacterial swarming algorithm	Chu et al.	[8]	Paddy Field Algorithm	Premaratne et al.	[48]
Firefly algorithm	Yang	[70]	Roach infestation algorithm	Havens	[25]
Fish swarm/school	Li et al.	[39]	Queen-bee evolution	Jung	[30]
Good lattice swarm optimization	Su et al.	[58]	Shuffled frog leaping algorithm	Eusuff and Lansey	[18]
Glowworm swarm optimization	Krishnanand and Ghose	[37], [38]	Termite colony optimization	Hedayatzadeh et al.	[27]
Hierarchical swarm model	Chen et al.	[5]	Physics and Chemistry based algorithms		
Krill Herd	Gandomi and Alavi	[22]	Big bang-big Crunch	Zandi et al.	[79]
Monkey search	Mucherino and Seref	[44]	Black hole	Hatamlou	[24]
Particle swarm algorithm	Kennedy and Eberhart	[35]	Central force optimization	Formato	[21]
Virtual ant algorithm	Yang	[77]	Charged system search	Kaveh and Talatahari	[34]
Virtual bees	Yang	[68]	Electro-magnetism optimization	Cuevas et al.	[13]
Weightless Swarm Algorithm	Ting et al.	[63]	Galaxy-based search algorithm	Shah-Hosseini	[53]
Other algorithms			Gravitational search	Rashedi et al.	[50]
Anarchic society optimization	Shayeghi and Dadashpour	[54]	Harmony search	Geem et al.	[23]
Artificial cooperative search	Civicioglu	[9]	Intelligent water drop	Shah-Hosseini	[52]
Backtracking optimization search	Civicioglu	[11]	River formation dynamics	Rabanal et al.	[49]
Differential search algorithm	Civicioglu	[10]	Self-propelled particles	Vicsek	[64]
Grammatical evolution	Ryan et al.	[51]	Simulated annealing	Kirkpatrick et al.	[36]
Imperialist competitive algorithm	Atashpaz-Gargari and Lucas	[2]	Stochastic difusion search	Bishop	[4]
League championship algorithm	Kashan	[32]	Spiral optimization	Tamura and Yasuda	[60]
Social emotional optimization	Xu et al.	[66]	Water cycle algorithm	Eskandar et al.	[17]

Figure 2.2: Nature-inspired algorithms included in the review of (Fister Jr. et al. 2013).

The classification of these algorithms varies depending on the criteria, and the literature does not provide an easy guideline to set it out. However, one approach for doing this is to consider its source of inspiration (Fister Jr. et al. 2013). Based on this criterion, the nature-inspired algorithms can be grouped as:

- *Swarm-intelligence-based*: these algorithms are related to the collective, emerging behaviour of multiple and interacting agents, who follow some simple rules. While each single agent may be considered as unintelligent, the whole system may show some self-organization behaviour and thus, can behave like some sort of collective intelligence. This type is among the most popular and widely used algorithms, and many have been developed by drawing inspiration from the collective behaviour of social insects (ants, termites, bees and wasps), as well as from other animal societies like flocks of birds or fish (Fister Jr. et al. 2013).

  Usually, in this type of algorithms the solutions are called “particles” and a specific solution value is known as its “position”, which during the optimisation process is continuously improved to get closer to the optimal position.

- *Bio-inspired-based*: are those algorithms inspired by biological phenomena or species, but without using collective swarm intelligence principles. This group represents the vast majority of all nature-inspired algorithms, being the Genetic Algorithms (GAs) the most relevant type. These GAs are an abstraction of biological evolution based on Charles Darwin’s theory of natural selection, which use genetic operators as their problem-solving strategy. Many variants of genetic algorithms have been developed so far and they have been applied to a wide range of optimisation problems (Yang 2014).

  The basic work-flow of GAs considers: (1) encoding the objectives functions; (2) defining a fitness function or selection criterion; (3) creating a population of individuals (here called “chromosomes”); (4) carrying out the evolution cycle by evaluating the fitness of all the chromosomes in the population, creating a new population (called “generation”) by performing genetic operators, and replacing the old population and iterating again using the new one; (5) decoding the results to obtain the solution of the problem (Yang 2014). The main genetic operators used in GAs are:

  - *Crossover*: is the main operator and is carried out by swapping one segment of one chromosome with the corresponding segment on another chromosome (the parents) at a random position (called single-point crossover). It can also occurs at multiple sites, which essentially swap the multiple segments with those on their corresponding chromosomes. Its main role is to provide mixing of the solutions and convergence in a subspace (Yang 2014).

- *Mutation*: it is achieved by flopping a randomly selected segment of the chromosomes, given certain probability. Its main role is to increase the diversity of the population and to provide a mechanism for escaping from a local optimum (Yang 2014).

- *Selection of the fittest (elitism)*: it is carried out by the evaluation of its fitness, which allows an individual to remain in the new generation if a certain threshold of the fitness is reached. Selection can also be fitness-based so that the reproduction of a population is fitness-proportionate, i.e., the individuals with higher fitness are more likely to reproduce (Fister Jr. et al. 2013).

In actual algorithms the interactions between these genetic operators can be very complex. However, the role of the individual components remains the same (Yang 2014).

- *Physics/Chemistry-based*: as their names suggest, these type of algorithms mimic certain physical and/or chemical phenomena, including for instance electrical charges, temperature changes, gravity or river systems. Within this group the most popular algorithm is Simulated Annealing (SA), which mimics the annealing process of metals, cooling and freezing it into a crystalline state with the minimum energy and larger crystal sizes, which reduces the defects in metallic structures (Fister Jr. et al. 2013; Yang 2014). The annealing optimisation process involves the control of temperature and its cooling rate (called the annealing schedule).

As can be seen in Figure 2.2 there are several proposed algorithms for solving optimisation problems, and the list is continuously increasing with newer proposals. However, this collection of alternatives is not always useful, since there may be some confusion and distraction in the research of meta-heuristic algorithms. On one hand, researchers have focused on important novel ideas for solving difficult problems. On the other hand, some researchers artificially invent new algorithms for the sake of publications, with little improvement and no novelty (Fister Jr. et al. 2013).

This research considers the implementation and testing of several nature-inspired meta-heuristic algorithms. The selection took into account:

- To include one algorithm of each main group, choosing the most representative type within the group (i.e. the most cited type in the literature).

- The capacity of the algorithm to handle multi-objective optimisation problems.

Based on the previous, a *Particle Swarm Optimisation* algorithm was selected as representative of the swarm-intelligence-based group; a *Genetic Algorithm* was selected as representative of the bio-inspired-based group; and a *Simulated Annealing* algorithm was

selected as the representative of the Physics/Chemistry-based group. In addition, a very recently published algorithm was also selected, in order to include in the research a representative of the these “brand-new” algorithms, which somehow are a manifestation of the dynamism that this research field has shown during the last years.

Sections 2.4.2 to 2.4.5 present a summarised description for these four selected algorithm, based on the articles on which they were originally published. Because of this and in order to avoid excessive and redundant references to the same authors, unless a different citation is explicitly included in the text, all the information must be considered as coming from those original articles.

2.4.2 Standard Particle Swarm Optimisation

The Particle Swarm Optimisation (PSO) algorithm (Eberhart and Kennedy 1995; Kennedy and Eberhart 1995) is a population-based global optimisation technique inspired by the social behaviour of bird flocks looking for corn (therefore, it belongs to the Swarm Intelligence class of algorithms). Since its publication is has received a surge of attention in the literature, given its flexibility, easy computational implementation (programming), low computational requirements, low number of adjustable parameters and efficiency.

Numerous variants of the original PSO algorithm have been proposed in the literature, aimed at improving performance or tackling specific optimisation problems. Usually researchers claim to have compared their “improved” version of PSO to the “standard” PSO algorithm, but the standard itself seems to differ between different studies.

Standard PSO (SPSO) was proposed in order to establish a common benchmark and reference point to assess the performance of the numerous PSO variants appearing in the literature (Clerc 2006; Clerc 2012). Until now there have been three version of SPSO, the last one being from 2011 (SPSO-2011).

Despite closely following the original PSO algorithm, SPSO-2011 includes several improvements based on recent theoretical developments. However, SPSO-2011 is not intended to be the best PSO variant on the market but only be considered as the reference level to be outperformed by newer PSO improvements.

The algorithm starts by creating a random population (called the swarm) where each candidate solution (called particles) stores its position (the actual value of the solution), its fitness value at that position, an initially random “velocity” vector which will be used to compute its next position, a “memory” vector that contains the best position found so far by the particle (called the previous best) and the fitness value of that previous best.

The particles in the swarm are related to each other using a topology, which defines the links between the particles or “who informs who”. When a particle is “informed” by another ones,

it means that the particle knows the previous best of the "informing" particles (called the neighbourhood) and that information will be used later in the process.

Then, at each iteration the velocity of each particle is re-calculated using a set of equations which combine: i) the particle's current position; ii) the particle's current velocity; iii) the particle's previous best; iv) the best "previous best" in the neighbourhood. Thereafter the particle is "moved" to a new position by applying the new velocity to it.

A confinement method is also applied to ensure that the new position is still inside the search space and finally the new fitness is calculated based on the new position. If the new fitness is better than the fitness of the "previous best" then the "previous best" and its fitness are replaced by a copy of this new particle and its new fitness. The algorithm can be stopped by applying one of the following criteria:

- If the fitness value on the optimum point is known, a maximum admissible error is defined. As soon as the absolute difference between this known fitness on the optimum point and the best one that has been found is smaller than this error, the algorithm stops.

- A maximum number of fitness evaluations is given in advance. As in Standard PSO the swarm size is constant, this is equivalent to a maximum number of iterations.

Figure 2.3 presents the pseudo code for the SPSO-2011 algorithm. Further details for understanding the PSO and SPSO algorithms can be found in the studies "*Standard Particle Swarm Optimisation*" (Clerc 2006; Clerc 2012), "*Particle Swarm Optimization*" (Kennedy and Eberhart 1995), "*A new optimizer using particle swarm theory*" (Eberhart and Kennedy 1995), and also by analysing the source code available in the online repository of this research.

for $i = 1$ to N **do** {for each particle in the swarm}
  Initialise particles' positions (X_i) and velocities (V_i)
  Initialise personal/previous best, $\vec{P}_i$, and local best, $\vec{G}$
end for
repeat
  for $i = 1$ to N **do**
    Update particle's velocity using equation 3
    Update particle's position using equation 1b
    if $f(\vec{X}_i) < f(\vec{P}_i)$ **then** {minimization of f}
      Update particle's best-known position $\vec{P}_i = \vec{X}_i$
      if $f(\vec{P}_i) < f(\vec{L})$ **then** {minimization of f}
        Update the neighbourhood's best-known position $\vec{L} = \vec{P}_i$
      end if
    end if
  end for
until [nr. of iterations (T) or tolerance error is met]

Figure 2.3: Pseudo code for the SPSO-2011 algorithm (from Zambrano-Bigiarini et al. 2013).

2.4.3 Non-dominated Sorting Genetic Algorithm II

The Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al. 2002) was proposed to improve the previous version of this heuristic, which over the years received some criticisms focused on its: i) high computational complexity of non-dominated sorting; ii) lack of elitism; and iii) the necessity for specifying the sharing parameter σ_{share} to ensure diversity (which according to the author was difficult of properly set). By contrast, NSGA-II uses a fast non-dominated sorting procedure, an elitist-preserving approach and a parameterless niching operator which ensures diversity of the candidate solutions.

Initially, a random parent population P_0 of size N is created and then it is sorted based on the non-domination. Each solution receives a fitness (or rank) equal to its non-domination level (where 1 is the best level, 2 is the next-best level, and so on). Thus, minimization of fitness is assumed. At first, a binary tournament selection, recombination and mutation operators are applied over P_0 to create an offspring population Q_0 of size N. From the parent and offspring populations a new combined population R_t of size $2N$ is formed, which is also sorted according to non-domination. Since all previous and current population members are included in R_t elitism is ensured.

After this the solutions belonging to the best non-dominated set F_1 are the best solutions in the combined population and must be emphasized more than any other solution. If the size of F_1 is smaller than N all its members are selected for the new population P_{t+1}. The remaining members of the population P_{t+1} are chosen from subsequent non-dominated fronts in the order of their ranking. Thus, solutions from the set F_2 are chosen next, followed by solutions from the set F_3 and so on until no more sets can be accommodated.

The new population P_{t+1} of size N is now used for selection, crossover and mutation to create a new population Q_{t+1} of size N. Both populations are combined to create a new set with $2N$ solutions and the non-dominated sorting procedure is repeated again to obtain a population P_{t+2} of size N. This main loop is then repeated as many times as needed until the satisfaction of an end criterion (for example, number of iterations). The pseudo code of the main loop of NSGA-II is shown in Figure 2.4.

The basic operations of NSGA-II and their worst-case complexities are as follows: 1) Non-dominated sorting is $O(M(2N)^2)$; 2) Crowding-distance assignment is $O(M(2N) \log (2N))$; 3) Sorting on crowded-comparison is $O(2N \log (2N))$. Therefore, the overall complexity of NSGA-II is $O(M(2N)^2)$ where N is the size of the population, and it is governed by the non-dominated sorting part of the algorithm.

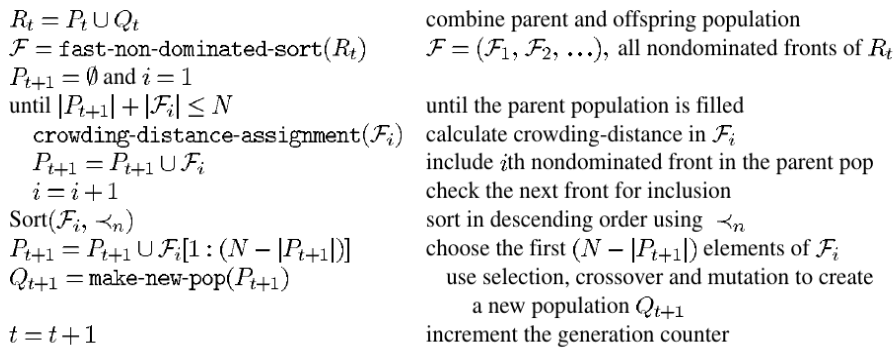


Figure 2.4: Pseudo code of the main loop of NSGA-II algorithm (Deb et al. 2002).

Further details for understanding the NSGA-II algorithm can be found in the study "*A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II*" (Deb et al. 2002) and also by analysing the source code available in the online repository of this research.

2.4.4 Archived Multi-Objective Simulated Annealing

The Archived Multi-Objective Simulated Annealing algorithm (AMOSA) (Bandyopadhyay et al. 2008) was proposed to improve the existing multi-objective implementations of simulated annealing technique, which in general do not consider Pareto dominance for accepting a new candidate solution as part of the final set. The algorithm has several stages that can be summarised as follows:

a) *Archive initialisation*: the algorithm begins initialising a random set of solutions which are refined by using a hill-climbing technique. A solution is accepted only if it dominates the previous one and the non-dominated candidates that were obtained are stored in the archive up to a Hard Limit (HL). In case the number of non-dominated solutions exceeds HL clustering is applied to restrict the size to that value. During this phase it is possible to get an archive of size one.

b) *Clustering solutions in the archive*: used to explicitly enforce the diversity of the non-dominated solutions. In general, the size of the archive is allowed to increase up to a Soft Limit (SL) higher than HL after which the solutions are clustered for grouping the solutions into HL clusters. For clustering the Single Linkage Algorithm is used (Jain and Dubes 1988). After clusters are obtained, the member within each cluster whose average distance to the other members is the minimum is considered as the representative member of the cluster. A tie is resolved arbitrarily. The representative points of all the clusters are thereafter stored in the archive.

c) *Amount of domination*: this concept is used by AMOSA to compute the acceptance probability of a new solution, depending on its fitnesses and the range of fitnesses of the solutions in the archive.

d) *Main AMOSA process*: one of the points (solutions), called *current-pt*, is randomly selected from the archive as the initial solution at temperature T_{max}. The *current-pt* is then "perturbed" (i.e. slightly modified by a random procedure) to generate a new solution called *new-pt*. The domination status of *new-pt* is checked with respect to the *current-pt* and the solutions in the archive. Then, based on the domination status between *current-pt* and *new-pt* three different cases may arise:

- *Case 1: current-pt* dominates the *new-pt* and *k* points from the archive dominate the *new-pt*. In this situation the *new-pt* is not accepted in the archive but it can be selected as *current-pt* with a given probability.

- *Case 2*: *current-pt* and *new-pt* are non-dominating with respect to each other. Now, based on the domination status of *new-pt* and members of the archive the following three situations may arise:

  1) *new-pt* is dominated by *k* points in the archive. Here the *new-pt* is not accepted in the archive but it can be selected as *current-pt* with a given probability (different to probability in Case 1).
  2) *new-pt* is non-dominating with respect to the points in the archive. In this case the *new-pt* is selected as the *current-pt* and also added to the archive. If the archive becomes overfull (SL is exceeded) clustering is performed to reduce the archive size to HL.
  3) *new-pt* dominates *k* points of the archive. In this case the *new-pt* is selected as *current-pt* and added to the archive. All the *k* dominated points are removed from the archive.

- *Case 3*: *new-pt* dominates *current-pt*. Based on the domination status of *new-pt* and the members of the archive three situations may arise:

  1) *new-pt* is dominated by *k* points in the archive. Here, the minimum of the difference of domination amounts between the *new-pt* and the points of the archive is computed. The point from the archive which corresponds to the minimum difference is selected as the *current-pt* with certain probability (different to the previous cases). Otherwise, the *new-pt* is selected as the *current-pt* but not included in the archive.
  2) *new-pt* is non-dominating with respect to the points in the archive. In this case *new-pt* is accepted in the archive and also as the *current-pt*. If the previous *current-pt* was part of the archive it is removed. If the archive becomes overfull clustering is performed to reduce its size to HL.

3) *new-pt* dominates *k* points of the archive. Here the *new-pt* is selected as the *current-pt* and also added to the archive, while all the *k* dominated points in the archive are removed.

The process is repeated *NumIter* times for each temperature, which is reduced to at each iteration using the cooling rate *α (α x temp*), until the minimum temperature (t_{min}) is reached. The process thereafter stops, and the resulting archive contains the final non-dominated solutions. Figure 2.5 presents the pseudo code for the AMOSA algorithm.

Set *Tmax*, *Tmin*, *HL*, *SL*, *iter*, α, temp=*Tmax*.
Initialize the *Archive*.
current-pt = random(*Archive*). /* randomly chosen solution from the *Archive* */
while (temp > *Tmin*)
  for (i=0; i< iter; i++)
    new-pt=perturb(*current-pt*).
    Check the domination status of *new-pt* and *current-pt*.
    /* Code for different cases */
    if (*current-pt* dominates *new-pt*) /* Case 1*/
      $\Delta dom_{avg} = \frac{\left(\sum_{i=1}^{k} \Delta dom_{i,new-pt}\right)+\Delta dom_{current-pt,new-pt}}{(k+1)}$.
      /* *k*=total-no-of points in the *Archive* which dominate *new-pt*, $k \geq 0$. */
      $prob = \frac{1}{1+exp(\Delta dom_{avg} * temp)}$.
      Set *new-pt* as *current-pt* with probability=*prob*
    if (*current-pt* and *new-pt* are non-dominating to each other) /* Case 2*/
      Check the domination status of *new-pt* and points in the *Archive*.
      if (*new-pt* is dominated by *k* ($k \geq 1$) points in the *Archive*) /* Case 2(a)*/
        $prob = \frac{1}{1+exp(\Delta dom_{avg} * temp)}$.
        $\Delta dom_{avg} = \frac{\left(\sum_{i=1}^{k} \Delta dom_{i,new-pt}\right)}{k}$.
        Set *new-pt* as *current-pt* with probability=*prob*.
      if (*new-pt* is non-dominating w.r.t all the points in the *Archive*) /* Case 2(b)*/
        Set *new-pt* as *current-pt* and add *new-pt* to the *Archive*.
        if *Archive-size* > *SL*
          Cluster *Archive* to *HL* number of clusters.
      if (*new-pt* dominates *k*, ($k \geq 1$) points of the *Archive*) /* Case 2(c)*/
        Set *new-pt* as *current-pt* and add it to the *Archive*.
        Remove all the *k* dominated points from the *Archive*.
    if (*new-pt* dominates *current-pt*) /* Case 3 */
      Check the domination status of *new-pt* and points in the *Archive*.
      if (*new-pt* is dominated by *k* ($k \geq 1$) points in the *Archive*) /* Case 3(a)*/
        Δdom_{min} = minimum of the difference of domination amounts between the *new-pt* and the *k* points
        $prob = \frac{1}{1+exp(-\Delta dom_{min})}$.
        Set point of the archive which corresponds to Δdom_{min} as *current-pt* with probability=*prob*
        else set *new-pt* as *current-pt*.
      if (*new-pt* is non-dominating with respect to the points in the *Archive*) /* Case 3(b) */
        Set *new-pt* as the *current-pt* and add it to the *Archive*.
        if *current-pt* is in the *Archive*, remove it from the *Archive*.
        else if *Archive-size*> *SL*.
          Cluster *Archive* to *HL* number of clusters.
      if (*new-pt* dominates *k* other points in the *Archive*) /* Case 3(c)*/
        Set *new-pt* as *current-pt* and add it to the *Archive*.
        Remove all the *k* dominated points from the *Archive*.
  End for
temp= α*temp.
End while
if *Archive-size* > *SL*
  Cluster *Archive* to *HL* number of clusters.

Figure 2.5: Pseudo code of the AMOSA algorithm (Bandyopadhyay et al. 2008).

The total complexity of AMOSA is defined by:

$$(SL + M + M \text{ x } SL) \text{ x } (TotalIter) + (TotalIter / (SL - HL)) \text{ x } SL^2 \text{ x } \log(SL)$$

where *M* is the number of objectives. Further details for understanding the AMOSA algorithm can be found in the study "*A Simulated Annealing-Based Multiobjective Optimization Algorithm: AMOSA*" (Bandyopadhyay et al. 2008) and also by analysing the source code available in the online repository of this research.

2.4.5 Multi-Objective Grey Wolf Optimiser

The Grey Wolf Optimizer (GWO) was proposed as a new nature-inspired algorithm mimicking the leadership hierarchy and hunting mechanism of grey wolves in nature (Mirjalili et al. 2014). The initial proposal was a single-objective optimiser but later it was adapted to create the Multi-Objective version (MOGWO)(Mirjalili et al. 2016).

In order to model the social hierarchy of wolves the fittest solution is considered as the alpha (α) wolf, the second and third best solutions are named beta (β) and delta (δ) wolves respectively, who lead the pack (omega (ω) wolves) toward promising regions of the search space, in order to find solutions close to the global optimum.

In MOGWO the "hunting" (i.e. the optimisation) is guided by α, β and δ, while the ω wolves "follow" these three wolves in the search for the global optimum (maximum or minimum). This means that at each iteration the "position" (i.e. the actual value of that solution at that moment) of each ω wolf is modified using a set of equations which take into account the "positions" of α, β and δ. Therefore, the new "position" of a ω wolf will be a consequence of its previous value, a random factor and the value (position) of the three pack leaders (three of the non-dominated solutions found so far).

The algorithm starts creating a set of random solutions (the first pack) and the three best obtained solutions are considered as the leaders. Then, for each omega wolf the position updating formulas are triggered. To perform multi-objective optimisation MOGWO utilises a simple storage unit called "archive" for storing the non-dominated Pareto optimal solutions. Figure 2.6 presents the pseudo code for the MOGWO algorithm.

During the course of iteration the non-dominated solutions obtained so far are compared against the archive residents and three possible cases can arise:

- *Case 1*: The new wolf is dominated by at least one of the wolves in the archive (including the leaders). In this case the solution is not accepted to enter to the archive.

- *Case 2*: The new wolf dominates one or more solutions in the archive. In this case the dominated solution(s) in the archive should be deleted from it and the new solution will be able to enter the archive.

- *Case 3*: If neither the new solution nor archive members dominate each other, the new solution should be added to the archive. If the archive is full a "grid mechanism" must

be first run to re-arrange the segmentation of the objective space and find the most crowded segment to omit one of its solutions. Then, the new solution is inserted to the least crowded segment, in order to improve the diversity of the final approximated Pareto optimal front.

```
Initialize the grey wolf population Xi (i = 1, 2, ..., n)
Initialize a, A, and C
Calculate the objective values for each search agent
Find the non-dominated solutions and initialized the archive with them
Xα=SelectLeader(archive)
Exclude alpha from the archive temporarily to avoid selecting the same leader
Xβ= SelectLeader(archive)
Exclude beta from the archive temporarily to avoid selecting the same leader
Xδ= SelectLeader(archive)
Add back alpha and beta to the archive
t=1;
while (t < Max number of iterations)
        for each search agent
                Update the position of the current search agent
        end for
        Update a, A, and C
        Calculate the objective values of all search agents
        Find the non-dominated solutions
        Update the archive with respect to the obtained non-dominated solutions
        If the archive is full
                Run the grid mechanism to omit one of the current archive members
                Add the new solution to the archive
        end if
        If any of the new added solutions to the archive is located outside the hypercubes
                Update the grids to cover the new solution(s)
        end if
        Xα=SelectLeader(archive)
        Exclude alpha from the archive temporarily to avoid selecting the same leader
        Xβ= SelectLeader(archive)
        Exclude beta from the archive temporarily to avoid selecting the same leader
        Xδ= SelectLeader(archive)
        Add back alpha and beta to the archive
        t=t+1
end while
return archive
```

Figure 2.6: Pseudo code for the MOGWO algorithm (Mirjalili et al. 2016).

The computational complexity of MOGWO is $O(MN^2)$ where N is the number of wolves in the pack and M is the number of objectives, which is is equal to the complexity of other well-known algorithms like NSGA-II.

Further details for understanding the MOGWO algorithm can be found in the study "*Multi-Objective grey wolf optimizer: A novel algorithm for multi-criterion optimization*" (Mirjalili et al. 2016) and also by analysing the source code available in the online repository of this research.

Chapter 3
Methods

3.1 Problem definition

Given an urban settlement, it is necessary to design an evacuation plan for its population in order to keep it safe in an emergency situation. Thus, the plan requires to allocate people from the locations where they are (working places or residential zones) to a limited number of safe areas, which have a total carrying capacity lower than the population that needs to be evacuated.

A possible first approach to handle this problem would be to allocate inhabitants to the closest available safe area. However, these places are not evenly distributed in the settlement and their capacity for accepting people also varies from one to another. Therefore, following this approach could lead to undesirable situations like overcrowding or even conflicts between people for getting a safe place to stay.

An alternative approach is to assign the population to the safe areas as equally as possible, in order to reduce the overcrowding and to ensure a better distribution for the whole community. Nevertheless, this approach implies that in some cases, certain people will move more than others even if they have a safe area close to them.

From the decision-makers' point of view there are two main objectives to deal with: on one hand, to optimise how people is distributed into the safe places; on the other hand, to optimise the total displacement (meters per person) required to evacuate the dangerous areas, since the lower the displacement the faster is the evacuation.

As can be deducted, these two objectives are conflicting with each other and must be optimised simultaneously to obtain a traded off solution for the problem. Also, these two objectives have spatial characteristics and hence the problem can be defined as a spatial multi-objective optimisation problem.

Consequently, the problem can be re-phrased as: the search for the best possible allocation scheme for an evacuated population into several safe areas, minimising the overcrowding on such areas as well as minimising the total displacement of the evacuated people.

3.2 Objectives functions

The identified objectives must be expressed in a mathematical way in order to be evaluated during the optimisation process. Despite the given problem is related with living entities (people) from an abstract point of view it can also be seen as a traditional transport-allocation case.

The formal expressions that were used in this research for each objective function are:

- Accumulated Distance (minimise): $$f_{distance} = \sum_{j=1}^{n} \sum_{i=1}^{m} d_{ij} p_{ij} \quad (3.1)$$

- Capacity Overload (minimise): $$f_{capacity} = \sum_{j=1}^{n} \left| \frac{\sum_{i=1}^{m} p_{ij}}{c_j} - 1 \right| \quad (3.2)$$

where m is the number of "points of origin" of people (working places or houses); n is the number of safe areas; d_{ij} is the distance between the ith point of origin and the jth safe area; p_{ij} is the population in the ith point of origin being evacuated to the jth safe area; and c_j is the capacity of the jth safe area for receiving people.

The Operational Research literature provides several models to handle this type of problems, and some of them have been previously applied to emergency management or evacuation situations (Gen et al. 2008; Saadatseresht et al. 2009; Zheng et al. 2015).

3.3 Study area

In February 2010 an earthquake of magnitude 8.8 on the Richter scale struck central Chile, followed by the biggest tsunami since 1960. As a consequence several cities in the coastal zones were affected producing human losses and material damage. One of the most impacted areas was Constitución, a small city in the Maule region with around 37,000 inhabitants, where the tsunami wave was estimated in 15 meters high.

During the last years the National Emergency Office (ONEMI) has been implementing risk maps and evacuation plans for coastal areas, but in many cases they only consider the delimitation of dangerous zones, without including yet the location of safe areas or analysing different allocation alternatives for the evacuated population.

In the case of Constitución most of the city is located in areas which would need to be evacuated if a new tsunami occurs, as can be seen in Figure 3.1 where the yellowed areas are the zones under risk of inundation by the tsunami and the coloured polygons are the blocks that need to be evacuated.

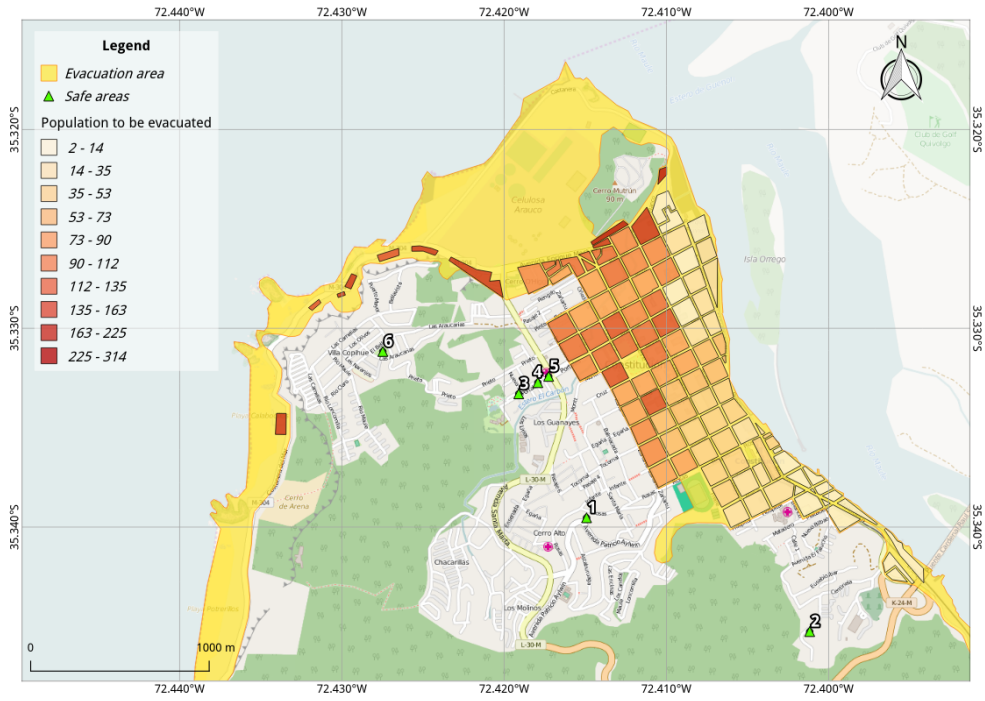


Figure 3.1: General view of the Constitución city (Maule region, Chile).

Constitución city is a good case study to test the proposed optimisation approach, since it presents the following critical aspects: i) A large population to be evacuated; and ii) Scarce and unevenly distributed safe areas, also very limited in their capacity for receiving people.

3.4 Performed tasks

In order to solve the given problem the following main stages were executed: i) Input data preparation, ii) Multi-Objective optimisation process, and iii) Visualisation of results, which are described in the following sections.

3.4.1 Input data preparation

The algorithms require two types of data to work. In the first case, related to the points of origin of the population, a table is needed to store, for each point of origin, a unique identifier, its coordinates, the population to be evacuated and the distance to each safe area following the shortest path. In the second case, related to the safe areas, a table is needed to store, for each

safe area, a unique identifier, its coordinates and its capacity for accepting people. The road network is also needed to calculate the shortest paths between the previous points.

The population data from the Chilean National Statistics Institute (INE) was aggregated to the level of census blocks, and therefore there was only one population value for each block. In order to represent a more realistic situation, the population of the blocks was divided in groups from six up to ten people (proportional to the whole population) and then these groups were randomly distributed within the blocks, like the actual houses (see Figure 3.2).

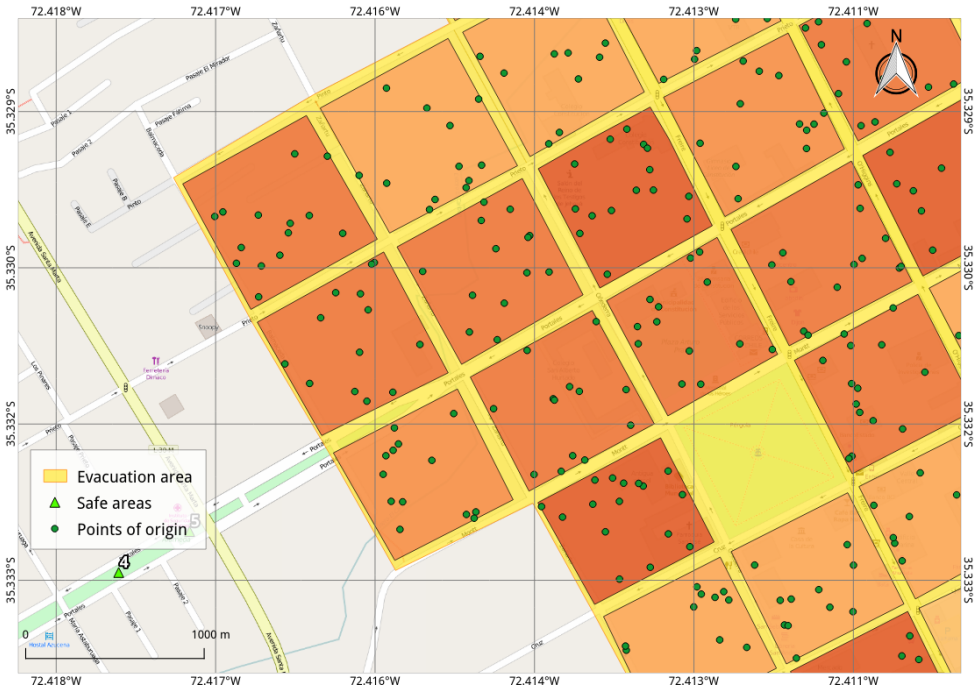


Figure 3.2: Distribution of points of origin at each block within the evacuation area.

Since the current evacuation plan developed by ONEMI only included “meeting points” but not clearly delimited areas, six polygons were created by using satellite imagery to delimit the open areas next to these “meeting points”. These polygons were considered as the actual safe areas and their capacity was estimated by calculating the area of the polygon and a safe standing crowd density of one people by square meter (Still 2014).

Finally, the shortest path from each point of origin to each safe area was calculated by using the Mapzen Public Routing API, utilising as input the street network provided by the OpenStreetMap project. All the previous tasks were executed using QGIS and Python scripts.

3.4.2 Multi-Objective optimisation process

After the input data was generated in the required format it was linked to a set of scripts in order to carry out the optimisation. The four selected algorithms were coded in Python

language from scratch, although for some specific functions complementary libraries were also used. These external modules were:

- Distributed Evolutionary Algorithms in Python (DEAP) (Fortin et al. 2012), used for creating the initial populations of candidate solutions and for Pareto dominance comparisons.
- Numpy & SciPy (Van Der Walt et al. 2011), used for clustering functions and arrays manipulation.

The link between the input data and the scripts can be done in two different ways (adjustable by changing only one parameter in the main script) and without any effect in the optimisation process:

- Remote option: the data can be read from a remotely hosted PostGIS database, which can also provide routing information.
- Local option: the data can be read from CSV and GeoJSON files locally hosted.

The multi-objective optimisation problem can also be solved considering two main situations: unconstrained (Case 1) or constrained (Case 2). In the first case the population of any point of origin can be allocated to any safe area. In the second case a limitation can be defined to restrict the search, in terms of a maximum distance (Case 2a) or a maximum number of safe areas (Case 2b).

For instance, a run of the algorithms constrained by a distance α produced solutions where the population at the *i*th point of origin was allocated only to safe areas within a distance lower or equal than α (by following the shortest path). On the other hand, a run constrained by β number of safe areas produced candidates solutions where the population at the *i*th point of origin was only allocated to the β closest safe areas (by following the shortest path). A particular case of the latter situation is when β has a value of 1, which indeed represents the simple approach of assigning people to the closest safe area, without taking into account the optimisation of both objective functions.

Each algorithm relies its operation on a set of tailor-made parameters which define the way they produce and optimise the candidate solutions. Although the proper setting of these algorithm-specific parameters is a matter of several studies and is not part of this research, each algorithm was configured taking into account:

- The parameters setting used in the original study of each algorithm.
- To produce an equivalent number of iterations and evaluations of the objective functions.

Since heuristics algorithms use randomness to create the initial set of candidate solutions and for exploring the search space, their output is always different in respect to the previous run.

In order to assess their average performance, reliability and repeatability, each algorithm was run five times, for the case that showed the best trade-off between the objective functions.

3.4.3 Visualisation of results

This stage considered two main goals: a) to visualise the population allocation scheme obtained as a result, and b) to visualise how the optimisation process was performed.

For the visualisation of the allocation scheme several KML files were created, showing the point of origin of the population and the safe area assigned to it. An external Python library (SimpleKML, http://www.simplekml.com) was used to create the KML files. This task was performed for each run and then the results were visualised in QGIS.

In order to analyse the optimisation process several data were collected from the output of each algorithm, considering:

- amount of solutions in the final 1st Pareto front
- optimisation of f_{capacity}
- optimisation of f_{distance}
- final Pareto fronts (f_{capacity} vs. f_{distance}).
- execution time.

3.5 Software used and legal considerations

The present research has been developed only using Open Source and Free Software, identified as follows:

- Operative system: 64-bit GNU/Linux (Kernel 4.5.2)
- Desktop GIS: QGIS 2.14.2 (Essen)
- Routing engine: Mapzen Public Routing API (based on Valhalla project)
- Streets network: OpenStreetMap project (updated to May 2016)
- Python interpreter: Python 2.7.10
- IDE: PyCharm Community Edition 2016.1.2
- Office suite: LibreOffice 5.1.3.1

All the runs were executed using the same hardware (Intel Core i7-3630QM @2.40 GHz, 8 GiB of RAM). The source code for each algorithm can be downloaded from the Git repository of the research, available in https://gitlab.com/felino/mooa-gis under the terms of the GNU Lesser General Public License (version 2.1).

The population data was obtained from INE on response to a formal request, and it was aggregated to the census blocks level and therefore specific individuals cannot be identified.

3.6 Assumptions

Some assumptions were done before executing the spatial optimisation:

- The points of origin of population were treated as equals, without differentiating between commercial, residential or working areas.
- The population at each point origin was assumed as invariable between day and night and therefore the obtained results do not consider these potential variations.

3.7 Conceptual diagram

The conceptual diagram in Figure 3.3 shows the summary of methods applied in this research.

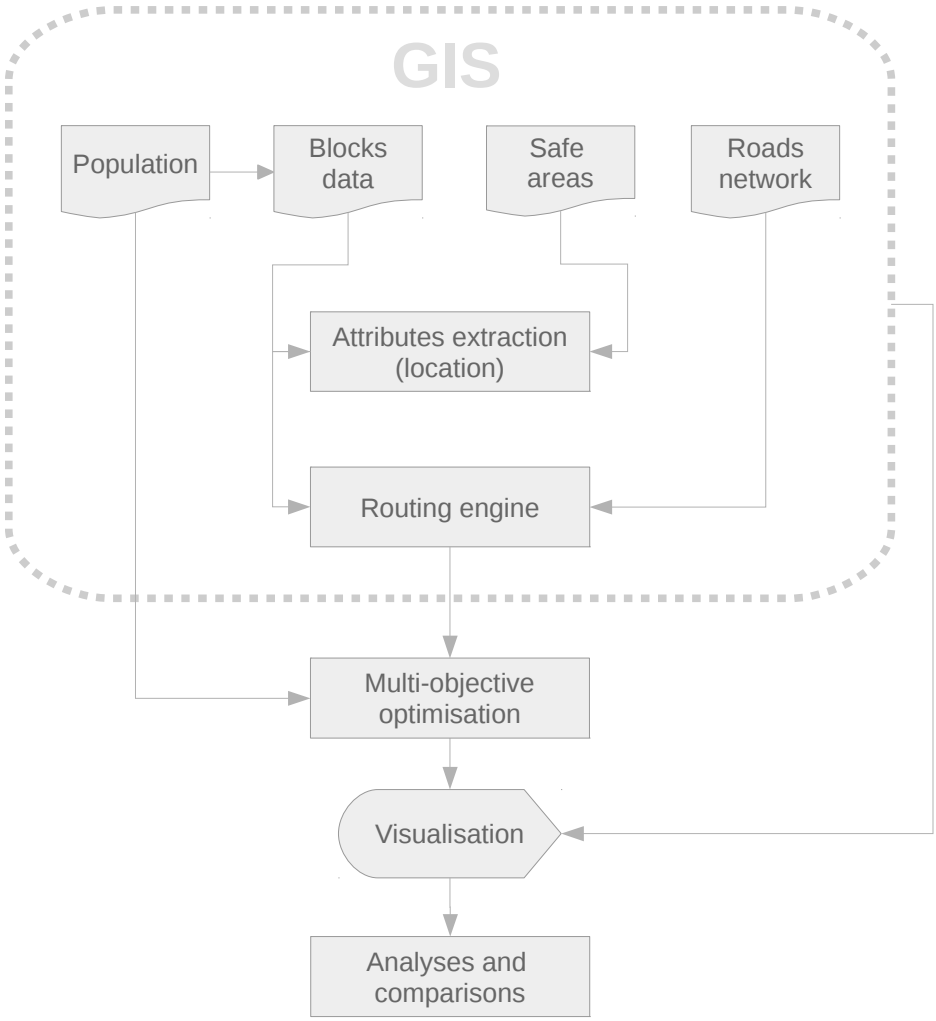


Figure 3.3: Conceptual diagram of methods applied for the research.

Chapter 4
Results

4.1 Overview

As mentioned earlier in the Methods chapter, the given multi-objective optimisation problem can be solved by using two main approaches, whether constraints are applied or not. For comparison purposes there is also a base case, in which people are allocated to the closest safe area without considering its capacity and therefore no actual multi-objective optimisation is applied.

The description of these cases is summarised in Table 4.1 and the results of the optimisation are presented in sections 4.4 to 4.7. Additionally, the whole output of the optimisation for each algorithm is given as charts in the Appendix of this report.

Table 4.1: Situations that may arise for solving the given spatial MOP.

Situation	Description
Case 0	**Allocation to the closest safe area**: corresponds to the base case and likely the first approach analysed when trying to solve this type of spatial problems. Only minimising the distance is taken into account and therefore, this case is equivalent to solve a single-objective problem
Case 1	**Unconstrained allocation:** people at any point of origin can be allocated to any safe area, no matter how far those areas are
Case 2a	**Constrained allocation by distance:** people at any point of origin can be allocated only to those safe areas within a given buffer. This constraint was set in 2,000 meters as maximum, based on the average distance between points of origins and safe areas
Case 2b	**Constrained allocation by number of safe areas:** people at any point of origin can be allocated only into the *n* closest safe areas. This constraint was set in *n*=3, which is half of the available safe areas in study area

4.2 Metrics

Two main metrics were defined in order to assess the performance of each algorithm:

- 1st Pareto front size: considers the amount of non-dominated solutions yielded by a single run of the algorithm. Since these solutions represent the trade-off between the objective functions, a higher amount will give to the decision maker a bigger set of good options to select from.
- Effectiveness of the optimisation: consists on how much each algorithm minimises the objective functions. The lower the values of the objective function the better the algorithm performs[1].

Additionally, the execution time, the diversity of the final 1st Pareto front and the repeatability of the algorithms were also analysed, although these aspects must be considered only as qualitative results. A higher diversity is valued, since a big set of solutions with low diversity will give the decision maker only a small set of actual options to select from. This, because even if the solutions are numerically different they are very similar to each other, and in practice they are not real alternatives for decision.

4.3 Parameters setting

As mentioned earlier, each algorithm have a set of parameters that defines the way they perform the optimisation. However, to test these parameters is out of the scope of this research and therefore their values were set based on the literature. Nonetheless, since each algorithm works in a different way, several pre-runs were executed in order to look for comparable conditions for them. From this exercise three common parameters were defined and their initial values are shown in Table 4.2.

Table 4.2: Initial values for three parameters set before running each algorithm.

Parameter	SPSO	NSGA-II	AMOSA	MOGWO
Search agents	50	200	120	50
Iterations	500	500	100	500
Storage size	n/a	n/a	50/60	50

The “search agents” is the amount of candidate solutions that are constantly being improved by each algorithm and it is a very algorithm-specific aspect. The “iterations” considers how many times the whole improvement cycle is executed. This value was lower for AMOSA since it works with nested loops that also depend on a “cooling rate”, resulting in less global iterations needed to reach comparable values. The “storage size” is the capacity of the

1 *f*capacity values are unitless, while the unit for *f*distance values is 10^7 (meters x person).

algorithm to collect the best solutions found so far, but it is only applicable to AMOSA and MOGWO since SPSO and NSGA-II do not consider this feature.

4.4 Allocation to the closest safe area

Table 4.3 presents the obtained results for the optimisation process considering Case 0.

Table 4.3: Results obtained from each algorithm for the Case 0 situation.

Case 0	1st front size	Minimum f_{capacity}	Minimum f_{distance}	Execution time
SPSO	1	2.7077323	0.9061308	00:03:44
NSGA-II	1	2.7077323	0.9061308	00:11:05
AMOSA	1	2.7077323	0.9061308	00:30:42
MOGWO	1	2.7077323	0.9061308	00:19:23

As can be seen, all the algorithms yielded a single solution with the same values, because in this case each point of origin only has one available safe area (the closest one, following the available pedestrian routes). Therefore, there is only one possible solution to be evaluated. Since the distribution of population is not considered in this case, the value of f_{distance} is the optimal for the given problem, and the value for f_{capacity} is the maximum for that function (the worst scenario of overcrowding, although not the theoretical worst scenario). The latter is produced because in Case 0 some safe areas are not used at all, as seen in Figure 4.1.

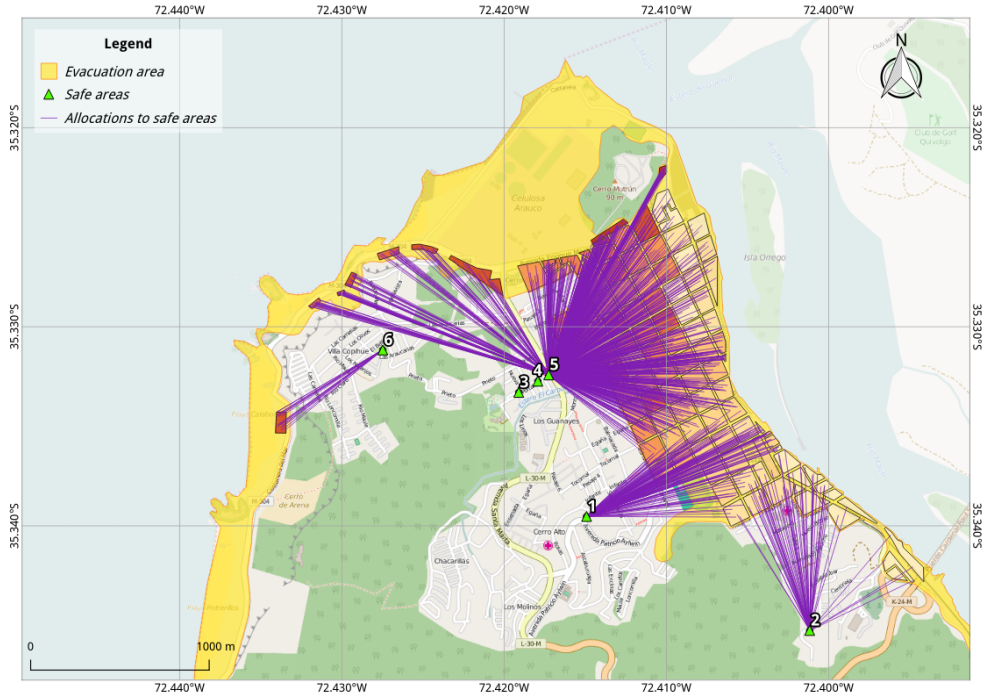


Figure 4.1: Allocation of population to the safe areas for Case 0 (safe areas 3 and 4 were not used at all, producing the worst scenario for $f_{capacity}$).

4.5 Unconstrained allocation

Figures 4.2 and 4.3 present the results obtained from each algorithm in terms of size of final 1st Pareto front and Effectiveness for Case 1.

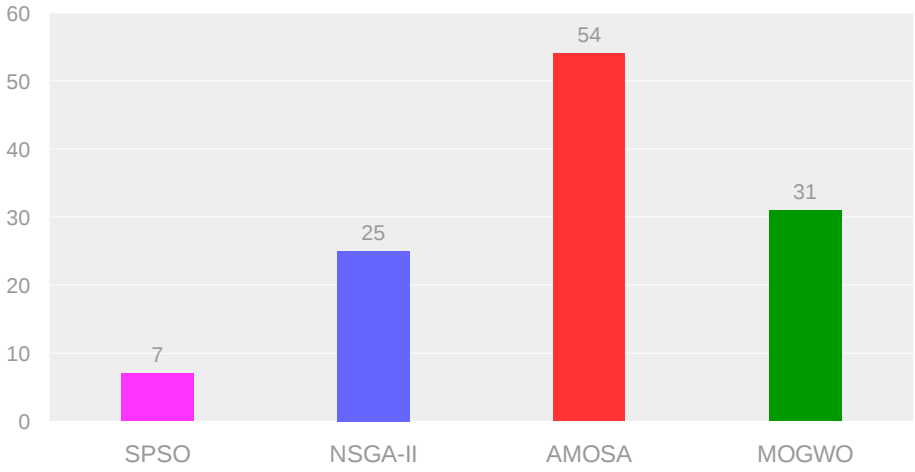


Figure 4.2: Solutions in the final 1st Pareto front of each algorithm for Case 1.

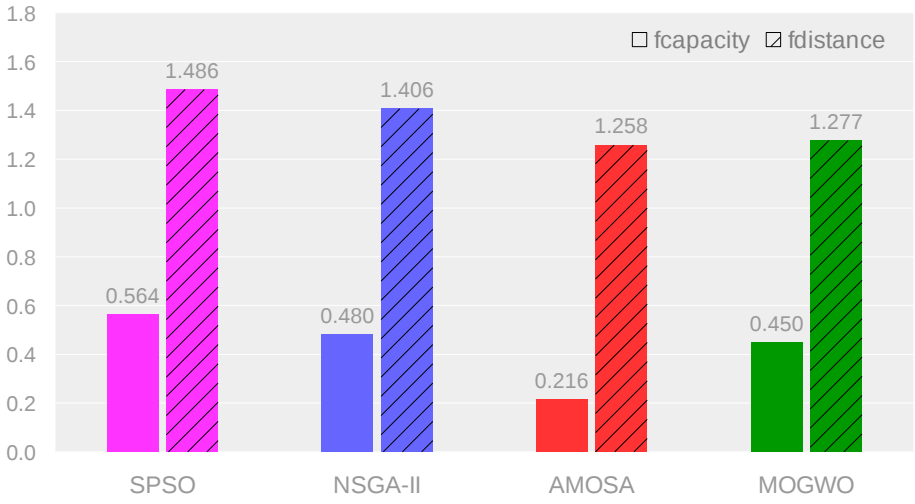


Figure 4.3: Effectiveness of optimisation for the Case 1.

AMOSA was the algorithm that delivered the highest amount of solutions and also the best one optimising both objective functions. In terms of diversity, the solutions yielded by AMOSA were evenly distributed along both axes, as can be seen in Figure 4.4.

It must be mentioned that in Case 1 the overall value of $f_{distance}$ is increased but the value of $f_{capacity}$ is highly optimised. Indeed, Case 1 is the best scenario for $f_{capacity}$ (although not the theoretical optimal) and also it is the worst scenario for $f_{distance}$. Because of this the allocation graphs look very crowded, since $f_{distance}$ was barely optimised (see Figure 4.5).

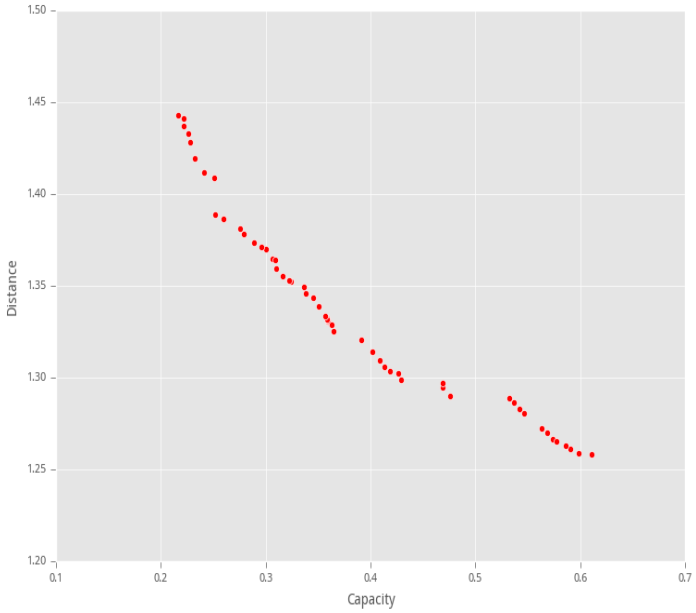


Figure 4.4: Final 1st Pareto front of AMOSA for Case 1.

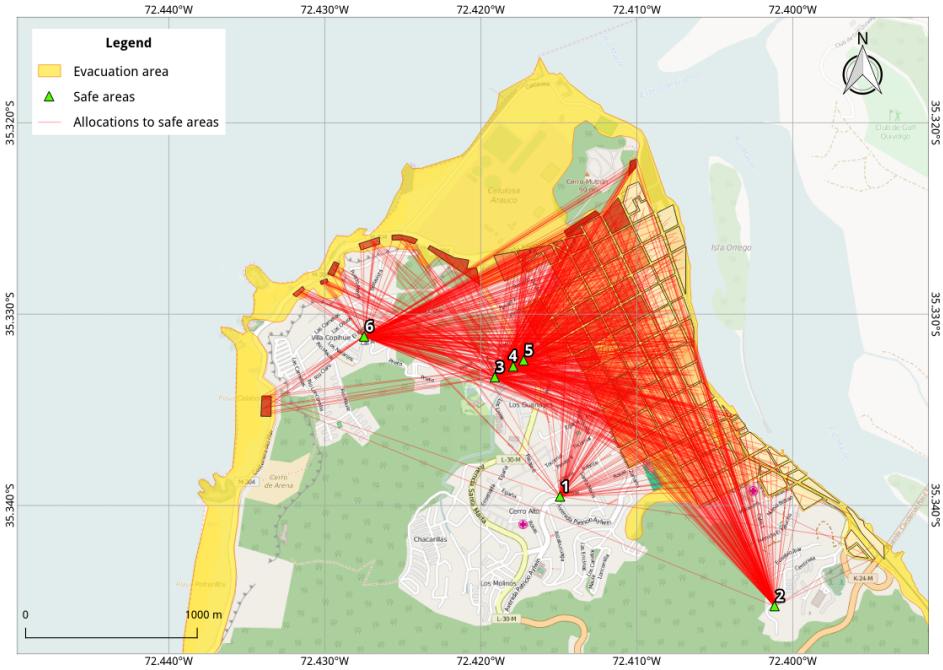


Figure 4.5: Allocations for Case 1, produced by the AMOSA algorithm.

Table 4.4 presents the summarised results of each algorithm for Case 1.

Table 4.4: Results produced by each algorithm for Case 1.

Case 1	1st front size	Minimum $f_{capacity}$	Minimum $f_{distance}$	Execution time
SPSO	7	0.5636374	1.4856007	00:12:11
NSGA-II	25	0.4804984	1.4064144	00:14:15
AMOSA	54	0.2162739	1.2582230	00:13:42
MOGWO	31	0.4498859	1.2769266	00:14:53

4.6 Constrained allocation by distance

Figures 4.6 and 4.7 present the results produced by each algorithm in terms of size of final 1st Pareto front and Effectiveness for Case 2a. AMOSA was again the best ranked algorithm although its results were very close to the values produced by the MOGWO algorithm.

AMOSA can store a variable number of non-dominated solutions (in this case study between 50 and 60) but MOGWO, instead, has a fixed limit of non-dominated solutions (in this case a maximum of 50). Therefore, even if it found more non-dominated solutions they were stored in the archive only after deleting another solution from it. Thus, the difference between both algorithms in relation to the size of the final 1st Pareto (54 vs. 50) can be considered as non-existing.

In terms of diversity, the final Pareto front of AMOSA was again evenly distributed with a high diversity. On the other hand MOGWO was adequately spread along the *x* axis ($f_{capacity}$) but its diversity was low for the *y* axis ($f_{distance}$), as can be seen in Figure 4.8.

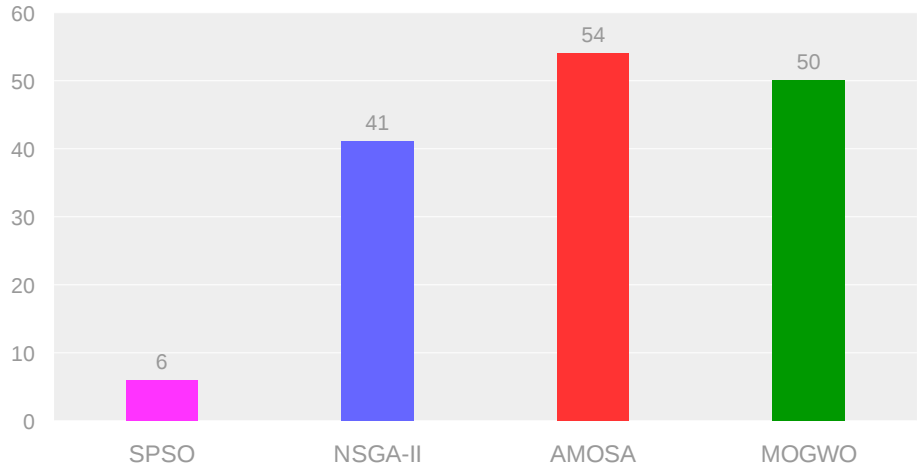


Figure 4.6: Solutions in the final 1st Pareto front of each algorithm for Case 2a.

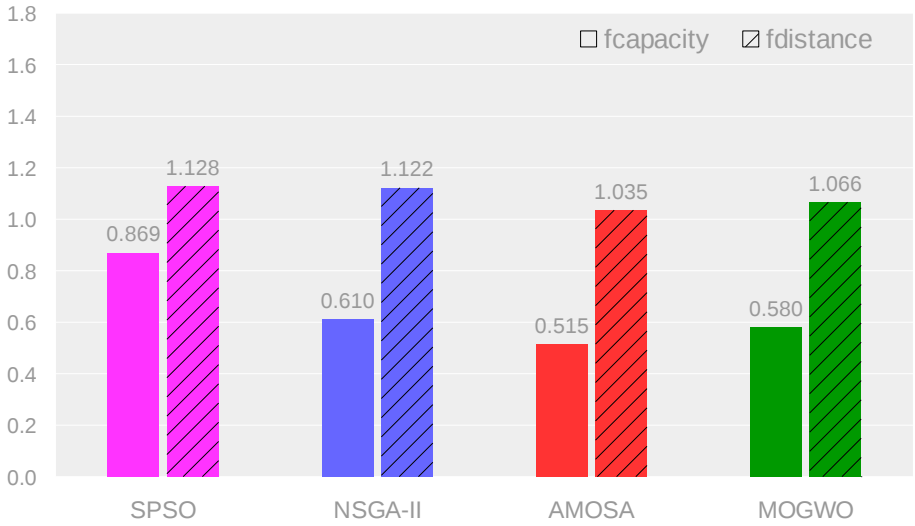


Figure 4.7: Effectiveness of optimisation for the Case 2a.

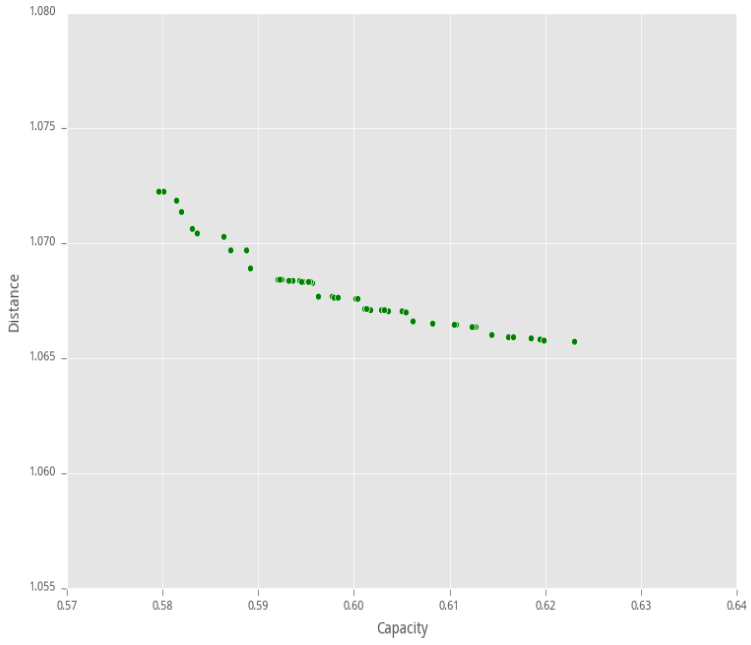


Figure 4.8: Final 1st Pareto front of MOGWO for Case 2a.

Because Case 2 is restricted by closeness, not all the points of origin have the same number of alternatives for allocation. Some points have only one or two potential destinations while others have five or six possible safe areas. This variability of solutions can be seen in figures 4.9 and 4.10. Even if AMOSA and MOGWO produced very similar optimised values, their allocation schema show several differences, as for example the allocation of people to safe areas 3 and 6.

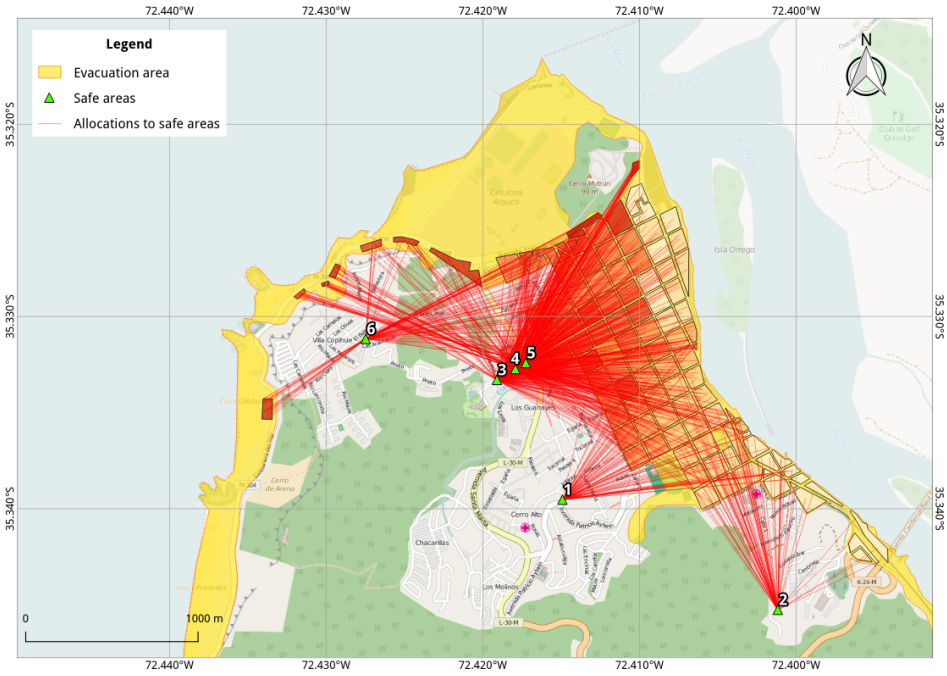


Figure 4.9: Allocations for Case 2a, produced by the AMOSA algorithm.

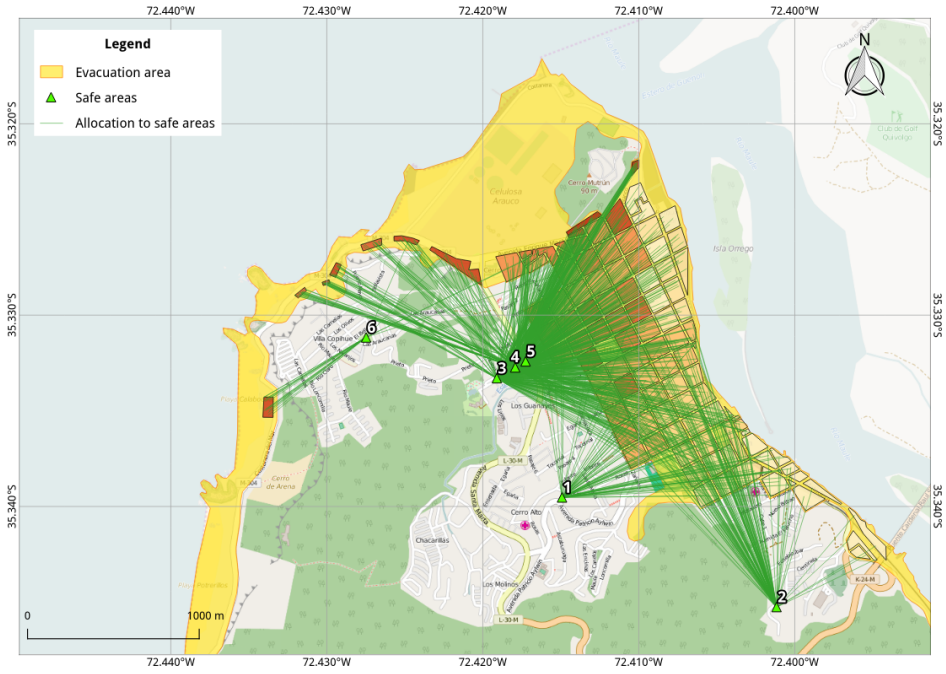


Figure 4.10: Allocations for Case 2a, produced by the MOGWO algorithm.

Table 4.5 presents the summarised results given by each algorithm for Case 2a. It can be noticed that the results show an increased level of trade-off between the objective functions, since f_{capacity} is increased and f_{distance} is decreased, in comparison with Case 1.

Table 4.5: Results produced by each algorithm for Case 2a.

Case 2a	1st front size	Minimum $f_{capacity}$	Minimum $f_{distance}$	Execution time
SPSO	6	0.8691519	1.1279685	00:12:41
NSGA-II	41	0.6098614	1.1217910	00:15:04
AMOSA	54	0.5148120	1.0346177	00:14:09
MOGWO	50	0.5795702	1.0657294	00:14:41

4.7 Constrained allocation by number of safe areas

Figures 4.11 and 4.12 present the results yielded by each algorithm in terms of size of final 1st Pareto front and Effectiveness for the Case 2b.

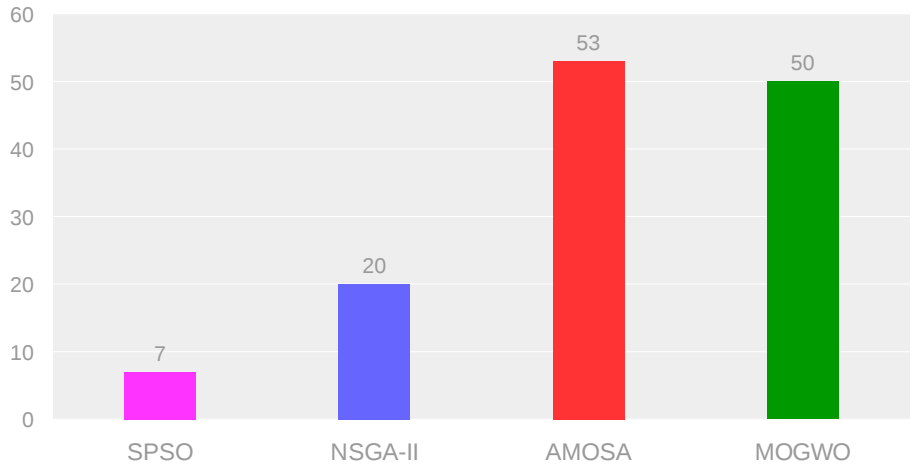


Figure 4.11: Solutions in the final 1st Pareto front of each algorithm for Case 2b.

In general terms AMOSA and MOGWO were once again the two best ranked algorithms for all the evaluated aspects, but this time NSGA-II and SPSO also produced comparable optimised values for $f_{distance}$.

As is shown in Figure 4.12, MOGWO got the best results on optimising $f_{capacity}$ and AMOSA got the best results on optimising $f_{distance}$. NSGA-II delivered almost similar values of optimisation for both objectives functions.

The number of solutions in the final 1st Pareto front is almost equal for AMOSA and MOGWO, and the diversity of their solutions is also well spread along both axes, as in cases 1 and 2a. However, the good results obtained by NSGA-II for $f_{distance}$ are impacted by a lack of diversity for this objective function, because a wide space in the *y* axis did not produced any solution, as can be seen in Figure 4.13.

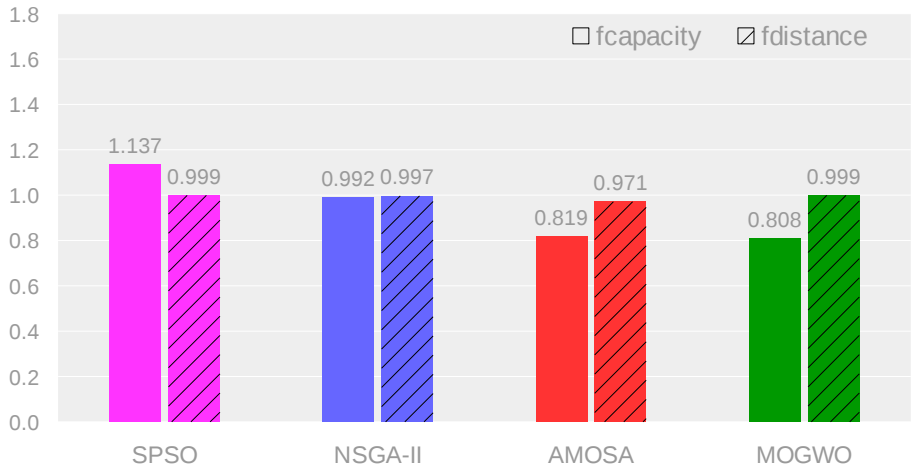


Figure 4.12: Effectiveness of optimisation for the Case 2b.

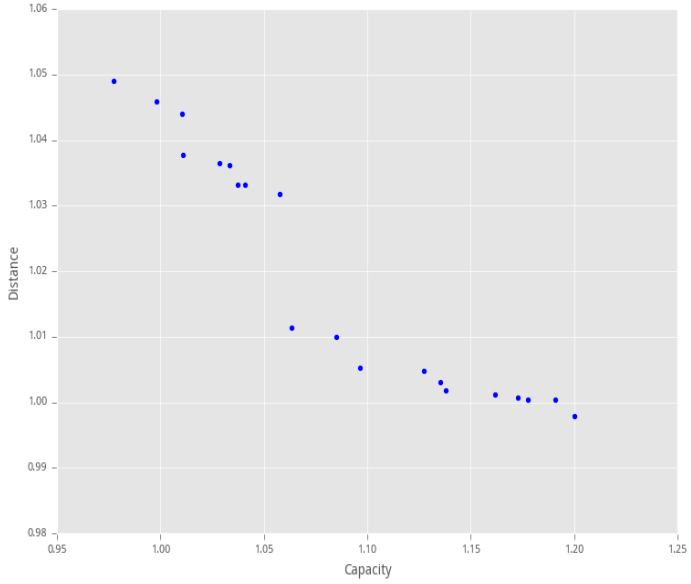


Figure 4.13: Final 1st Pareto front of NSGA-II for Case 2b.

In Case 2b every point of origin has the same number of alternatives to be allocated to a safe area, and consequently there is a high variability of potential solutions, and also this potential variability is shared by every point of origin. This circumstance produces a better distribution of population among the safe areas and also a reduction in the total distance. This can be observed in figures 4.14 and 4.15 where the lines connecting safe areas and points of origin look less crowded than the corresponding figures for previous cases.

Table 4.6 presents the summarised results given by each algorithm for Case 2b, which produced the most balanced set of solution so far (both objectives functions were highly optimised).

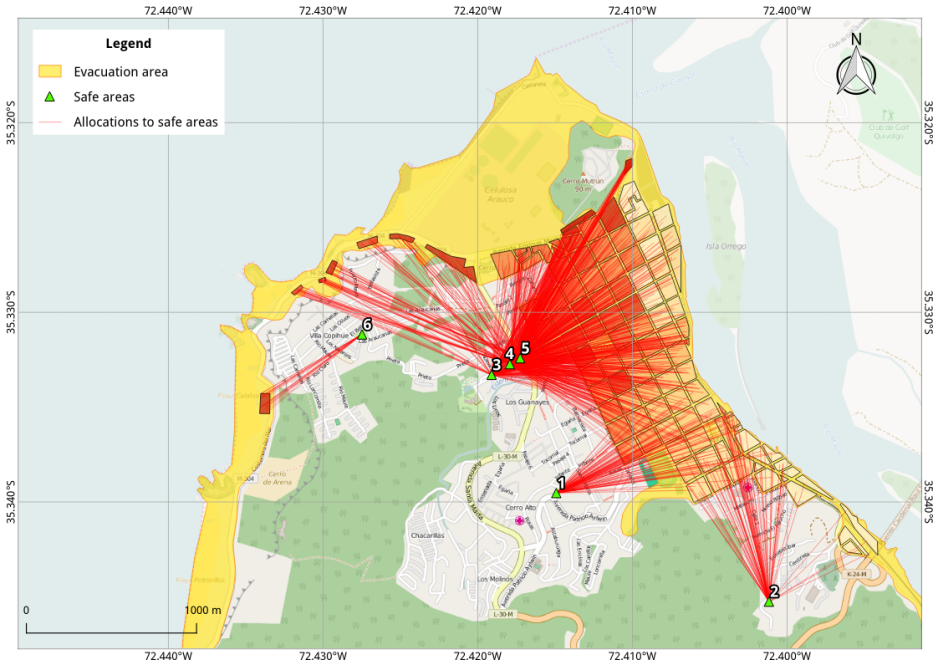


Figure 4.14: Allocations for Case 2b, produced by the AMOSA algorithm.

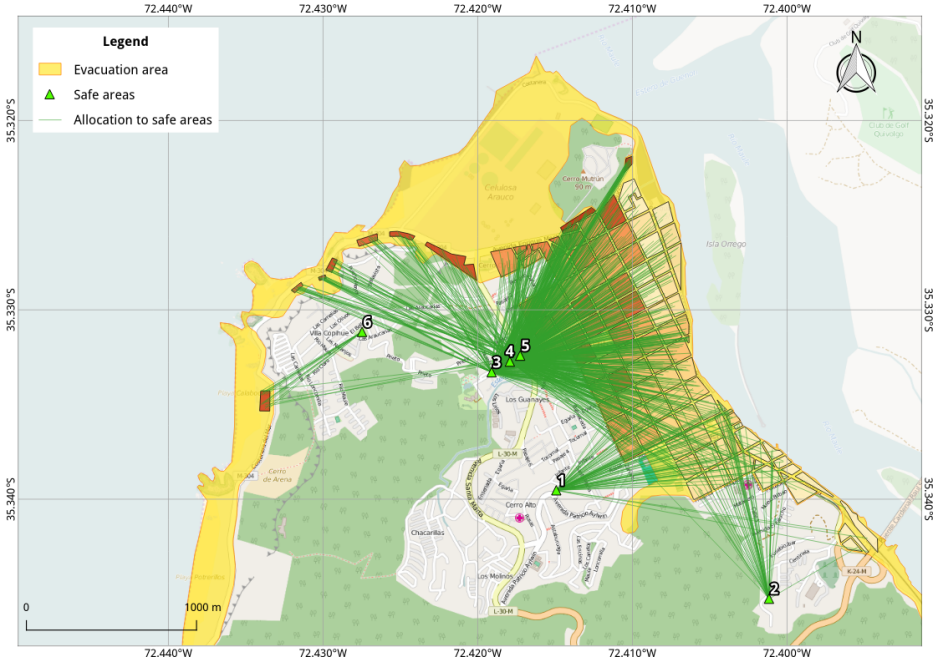


Figure 4.15: Allocations for Case 2b, produced by the MOGWO algorithm.

Table 4.6: Results produced by each algorithm for Case 2b.

Case 2b	1st front size	Minimum $f_{capacity}$	Minimum $f_{distance}$	Execution time
SPSO	7	1.1365114	0.9993371	00:15:17
NSGA-II	20	0.9919347	0.9967724	00:20:24
AMOSA	53	0.8190338	0.9709484	00:28:01
MOGWO	50	0.8077061	0.9991736	00:17:24

4.8 Repeatability test

The average performance and reliability of the algorithms were tested by running them five times for Case 2b, since this case showed the best trade-off between the objective functions. Due to the randomness of the algorithms the truncated mean was calculated (removing the highest and lowest values) to eliminate potential outliers. The results are shown in tables 4.7 to 4.10.

Table 4.7: Results of the repeatability test for SPSO algorithm in Case 2b.

SPSO	**1st front size**	**Minimum $f_{capacity}$**	**Minimum $f_{distance}$**	**Execution time**
Run 1	7	1.1365114	0.9993371	00:15:17
Run 2	3	1.1081282	1.0014307	00:07:48
Run 3	4	1.0920146	0.9994478	00:07:51
Run 4	7	1.1305858	0.9991321	00:08:01
Run 5	8	1.0876462	0.9971309	00:08:09
Truncated mean	6	1.1102429	0.9993057	00:08:00

Table 4.8: Results of the repeatability test for NSGA-II algorithm in Case 2b.

NSGA-II	**1st front size**	**Minimum $f_{capacity}$**	**Minimum $f_{distance}$**	**Execution time**
Run 1	20	0.9919347	0.9967724	00:20:24
Run 2	20	0.9772875	0.9979001	00:15:16
Run 3	20	0.9842009	0.9985792	00:15:15
Run 4	31	0.9949040	0.9979325	00:18:47
Run 5	21	1.0106955	0.9991129	00:15:36
Truncated mean	20	0.9903465	0.9981373	00:16:33

Table 4.9: Results of the repeatability test for AMOSA algorithm in Case 2b.

AMOSA	**1st front size**	**Minimum $f_{capacity}$**	**Minimum $f_{distance}$**	**Execution time**
Run 1	53	0.8190338	0.9709484	00:28:01
Run 2	58	0.8234791	0.9678276	00:20:29
Run 3	54	0.8369365	0.9690385	00:20:28
Run 4	58	0.8287831	0.9693966	00:22:45
Run 5	57	0.8227421	0.9662024	00:22:51
Truncated mean	56	0.8250014	0.9687542	00:22:02

Table 4.10: Results of the repeatability test for NSGA-II algorithm in Case 2b.

MOGWO	**1st front size**	**Minimum $f_{capacity}$**	**Minimum $f_{distance}$**	**Execution time**
Run 1	50	0.8077061	0.9991736	00:17:24
Run 2	31	0.8068500	0.9978106	00:12:53
Run 3	50	0.8220926	0.9960669	00:12:03
Run 4	49	0.8801099	0.9965750	00:12:24
Run 5	28	0.8347703	0.9987754	00:13:05
Truncated mean	43	0.8215230	0.9977203	00:12:47

All the algorithms showed good repeatability in terms of amount of solutions and execution time. If equations (1) and (2) in section 3.2 are applied to the dataset used for this research, the actual range of each function can be calculated, as shown in Table 4.11. Compared to these ranges, the maximum variations in the optimisation values between runs were also small, and even in the worst cases these variations were only 2.76% for f_{capacity} (MOGWO) and 0.27% for f_{distance} (AMOSA), as shown in Table 4.12.

Table 4.11: Actual range of values for each objective functions.

Objective function	Actual minimum	Actual maximum
f_{capacity}	0.0501289	2.7077323
f_{distance}	0.9061308	2.6857035

Table 4.12: Variations in the optimisation values between runs.

Algorithm	Variation f_{capacity}	Variation f_{distance}
SPSO	1.84%	0.24%
NSGA-II	1.26%	0.13%
AMOSA	0.67%	0.27%
MOGWO	2.76%	0.17%

4.9 Comparison against the extreme values

Finally, in order to know the net optimisation rate provided by each algorithm, a comparison was done between their average results (tables 4.7 to 4.10) and the minimum and maximum values potentially reachable (Table 4.11). Table 4.13 shows the average optimisation of each algorithm, where the “Actual maximum” was considered as 0% of optimisation and the “Actual minimum” was the 100% optimised situation.

Table 4.13: Percentage of optimisation of the objective functions by each algorithm.

Algorithm	f_{capacity}	Optimisation f_{capacity}	f_{distance}	Optimisation f_{distance}
SPSO	1.1102429	60.11%	0.9993057	94.76%
NSGA-II	0.9903465	64.62%	0.9981373	94.83%
AMOSA	0.8250014	70.84%	0.9687542	96.48%
MOGWO	0.8215230	70.97%	0.9977203	94.85%

For f_{capacity} the best net optimisation rate was obtained once again by MOGWO and AMOSA, with NSGA-II and SPSO optimising around 5% and 10% less this objective, respectively. However, for f_{distance} the net performance of all the algorithms was very similar (with a slightly higher performance for AMOSA) and could be considered as equally good.

Chapter 5
Discussion

5.1 Overview

As presented in Chapter 1, the objectives of this research were focused on three main aspects: 1) the algorithms' performance on optimising the given multi-objective spatial problem, 2) the potential relation between the type of algorithm and the obtained results, and 3) the comparison between the results yielded by the algorithms and a raw base case without multi-objective optimisation. Consequently, the following discussion is presented in such order, along with a section proposing further developments and research opportunities.

5.2 Algorithm performance

In terms of effectiveness, it can be said that all the algorithms performed the optimisation in a consistent way, and no results were obtained that could suggest that some of them was trapped in a local minimum. While for some cases the differences were minor (like Case 2b), the overall results indicate that AMOSA and MOGWO were the most effective ones, consistently ahead of SPSO and NSGA-II.

In general, the worst results were achieved by SPSO, but even for this algorithm the optimisation rates were around 60% for f_{capacity} and about 95% for f_{distance}, if compared against the base case. At a first sight these results could be contradictory with the extended opinion available in the literature about the flexibility and efficiency of PSO-based algorithms. However, this lower performance could be explained by the definition of the algorithm itself, which is not intended to be the best available PSO variant but a common benchmark and reference point to assess the performance of the numerous PSO variants (Clerc 2006; Clerc 2012).

Regarding the amount of non-dominated solution delivered by each algorithm, which in practice represents how many good alternatives the decision-makers will have to make a selection from, AMOSA and MOGWO had the best performance if compared with SPSO and NSGA-II. This superior performance is also seen when analysing the size of the final 1st

Pareto fronts, since AMOSA and MOGWO showed continuous and well spread solutions for all the cases that were analysed.

Based on the comparison of the results between the different cases, it can be inferred that the performance of the algorithms is highly influenced by the constraints definition, and that the best multi-objective optimisation is achieved when such constraints are less restrictive and, therefore, all the points of origin have equivalent chances to be allocated to a safe area.

Case 2b was considered as the most representative situation since it showed the best trade-off relation between the objective functions. But despite this, different scenarios can also be analysed and even selected by the decision-makers as more realistic cases, if their particular interests and context demand it.

The repeatability test (tables 4.7 to 4.10) showed consistency between the results of each algorithm for several runs in all the proposed metrics (1st front size, Minimum f_{capacity}, Minimum f_{distance} and Execution time). This consistency is coherent with the description of the algorithms given by their authors (Deb et al. 2002; Bandyopadhyay et al. 2008; Clerc 2012; Mirjalili et al. 2016), and could be considered as evidence of the reliability of these techniques.

5.3 Relation between type of algorithm and solutions quality

As indicated in Chapter 2 (section 2.4.1), the algorithms used in this research were selected considering the main types of nature-inspired heuristics, as well as the most recently published ones. The representatives of the groups were: a) Swarm-intelligence-based algorithms: SPSO; b) Bio-inspired-based algorithms: NSGA-II; c) Physics/Chemistry-based algorithms: AMOSA; and d) Recently published algorithms: MOGWO (April 2016).

A comparison between them reveals that for the given problem some algorithms consistently performed better than another for the defined metrics and data. However, these quantitative results should be treated carefully since the magnitude of such differences is not significant enough to claim that one type of heuristic is clearly superior than other. If compared against the base case, it could be said that all the applied heuristics optimised the problem in an acceptable manner. SPSO, which delivered the “worst” overall results, obtained a very high optimisation rate for f_{distance} which is even numerically higher than the one obtained by AMOSA, the overall “best” optimiser. The biggest differences are related to f_{capacity} but even in this case, the results of SPSO and NSGA-II cannot be consider as deficient from a qualitative point of view.

This lack of clear differences rises the question if the quality of solutions depends on the type of algorithm used (for example swarm intelligence vs. simulated annealing) or if what defines the performance is how a specific implementation of an algorithm handles the critical aspects

of the optimisation process. Coincidently, AMOSA and MOGWO explicitly include new ways for ensuring diversity (clustering and hypercube segments density), storing the best solutions found so far (archives) and properly exploring the search space. This characteristics are not present in SPSO and NSGA-II and the superior performance of AMOSA and MOGWO could be connected to these implementation improvements, more than to the type of algorithms they belong to. In other words, from the obtained results it is not possible to state that one of the studied types of algorithm is the best for solving spatial problems in a GIS-MCDA context.

This perception is somehow supported by the “No free lunch” theorem, which in simple terms states that any two algorithms are equivalent when their performance is averaged across all possible problems, and that matching algorithms to problems gives higher average performance than applying a fixed algorithm to all the problems (Wolpert and Macready 1997).

This situation rises the inconvenience of assuming a specific type of algorithm as a “flagship” for solving spatial multi-objective optimisation problems, which however seems to be the way followed so far in the GIS-MODA field when reviewing the literature (most of the available research has been focused on genetic algorithms, specifically NSGA-II). Instead, an alternative strategy for the future could be to have an extensible set of algorithms, ready to be applied to different spatial multi-objective optimisation problems. By doing so the GIS tasks could be focused on the selection and interpretation of the results more than on the comparison of the heuristics.

5.4 Comparison against the base case

As presented in Chapter 4, the base case for the given spatial problem would be to allocate each point of origin to the closest safe area, without considering a multi-objective optimisation. In such case, the distribution of population among the safe areas would produce a highly crowded situation in some of those areas and not all them would be properly used.

The results of the research and particularly Table 4.13 show a high optimisation rate for both objectives, mainly for f_{distance} with all the results around 95% of effectiveness. Regarding the optimisation of f_{capacity}, even if the optimisation rates were lower than for the previous objective function they are still high (around 65% in average and more than 70% for the best cases), and for the study area used in this research these values can be limited by the specific location of the safe areas. In this sense, for an evacuation planning analysis these heuristic techniques could be used not only for assessing different allocations schema, but also for designing and selecting better safe areas, by comparing the overall optimisation rates against different safe areas availability.

An alternative approach to deal with this type of spatial problems, without using multi-objective optimisation, could be to allocate the population only considering the spatial

distribution of the safe areas and points of origin, for example, by using GIS analysis tools like Voronoi diagrams. However, this approach could yield distorted results, since the evacuation plan should consider feasible evacuation routes and the Voronoi approach does not take this aspect into account, but only the Euclidean distance between the points.

To overcome these limitations several corrections can be applied, like manually modifying a particular allocation when a safe area is not actually reachable from a point of origin. Nevertheless, these corrections could introduce a certain degree of bias into the analysis (they require to detect each allocation problem), and for complex situations (like large cities with complex streets networks) this task could be very time consuming and the amount of final alternatives given to the decision maker could be reduced.

By using heuristics the bias probability can be limited and reduced mostly to the definition of the constraints needed for running the optimisation (like the ones applied in this research), and based on the obtained results a good-enough set of solutions can be presented to the decision-makers. In other words, the use of these multi-objective optimisation techniques helps to transform the analysis from an *ex-ante* to an *ex-post* situation, focusing the decision process on actual alternatives where the trade-off between the objectives is known in advance, and therefore the decisions can be made in a more transparent and objective way.

The optimisations were done in an average execution time of about 15 minutes, which can be considered as very good and promising, if compared with the time potentially needed to perform similar analyses by using non-heuristic GIS-MCDA approaches.

5.5 Suggestions for future development

Some aspects may be proposed as future development and study opportunities, based on the literature review, the obtained results and some learned lesson from this research. The main aspect is related to the standardisation of the GIS-MCDA studies, in both the terminology used and the methods. On the available literature it is possible to find several ways to refer to different techniques under similar “umbrella” terms, as well as different terms to name the same technique, which may be confusing.

Also, the definition of a standard set of spatial problems to test the performance of the heuristics techniques in a GIS context would be desirable. This practice is commonly done in the evolutionary computation domain (for example, the set of benchmark functions designed for the annual IEEE Congress on Evolutionary Computation (http://www.ieee.org)), and would allow to verify if the algorithms have an average performance or if their efficiency is problem specific.

Another development area could be to consider the dynamic characteristics of real-world spatial problems during the multi-criteria analysis. Specifically for evacuation planning

situations, examples of this dynamism could be the variations in the distribution of population between day and night, working days and weekends, commercial and residential areas, and maybe most importantly, the uncertainty of human behaviour during emergency situations.

Further studies could also be done to increase the available research on optimising spatial problems with three or more simultaneous objectives, as well as more types of constraints which usually are inherent to real-world problems. To consider the dynamic update of the allocation scheme is another research opportunity (for example, what if one of the safe areas is not accessible due to emergency itself?), what could be addressed by using crowd-sourced data and mobile technologies.

Finally, most of the current implementations of GIS-MCDA are only based on the multi-attribute approach and therefore integration between these systems and the heuristic approach could lead to more robust GIS-based applications, able to handle a wider range of spatial problems and strengthening the role of GIS as decision-making support tools. The use of open standards as well as open software should be considered as a base concept, since they favour interoperability and can be implemented in situations where the access to proprietary software could be a *de facto* restriction (for example, for developing economies).

Chapter 6
Conclusions

From the performed research it is possible to conclude that the four selected algorithms were effective on solving the given spatial multi-objective optimisation problem, related to an evacuation planning situation for the Constitución city in the Maule region, Chile. The algorithms yielded high rates of optimisation for both objective functions, and for the best cases they also delivered a good and constant number of non-dominated solutions. Their average performance was consistent across several runs, which can be understood as a sign of maturity and reliability of these techniques. Generally AMOSA and MOGWO performed best, but the differences in the results are not considered as significant, and therefore, it is not possible to state that one type of algorithm is clearly superior than others for solving the given spatial multi-objective problem.

Alternative approaches to deal with this type of spatial multi-objective optimisation problems without using heuristics can be executed by using traditional GIS analysis tools. However these methods could face several limitation in terms of time consumption, amount of equivalent solutions given to the decision-makers and bias introduction. The use of optimisation heuristics could help to reduce these limitations, focusing the decision-making on actual, transparent and more diverse alternatives.

Several suggestions for further development are given regarding the standardisation of GIS-MCDA terminology; the standardisation of how heuristic are applied to optimise spatial multi-objective problems; the inclusion of temporal and dynamic aspects of real-world problems during the optimisation process; the use of these techniques to optimise many-objectives spatial problems and the integration of this heuristic approach with the current implementations of GIS-MCDA, in order to build more robust GIS-based applications.

Finally, it is considered that multi-objective optimisation algorithms are a reliable method for handling and solving spatial problems with multiple and conflictive objectives, and that future and more practical implementations in this field will strengthen the capacities of GIS as a multi-criteria decision-making support tool.

References

Bandyopadhyay, S., S. Saha, U. Maulik, and K. Deb. 2008. A Simulated Annealing-Based Multiobjective Optimization Algorithm: AMOSA. *IEEE Transactions on Evolutionary Computation* 12: 269–283. doi:10.1109/TEVC.2007.900837.

Boerboom, L. 2012. Implementation, challenges and future directions of integrating services from the GIS and decision science domains; a case of Distributed Spatial Multi-Criteria Evaluation. *OSGeo Journal* 10: 6.

Bonyadi, M. R., and Z. Michalewicz. 2014. SPSO 2011: Analysis of Stability; Local Convergence; and Rotation Sensitivity. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, 9–16. GECCO '14. New York, NY, USA: ACM. doi:10.1145/2576768.2598263.

Carver, S. J. 1991. Integrating multi-criteria evaluation with geographical information systems. *International Journal of Geographical Information Systems* 5: 321–339. doi:10.1080/02693799108927858.

Clerc, M. 2006. *Particle swarm optimization*. London ; Newport Beach: ISTE. Retrieved from http://dx.doi.org/10.1002/9780470612163

Clerc, M. 2012. Standard Particle Swarm Optimisation. Retrieved from https://hal.archives-ouvertes.fr/hal-00764996/document

Coello, C. 2009. Evolutionary multi-objective optimization: some current research trends and topics that remain to be explored. *Frontiers of Computer Science in China* 3: 18–30. doi:10.1007/s11704-009-0005-7.

Coello, C., G. Lamont, and D. Van Veldhuizen. 2007. *Evolutionary algorithms for solving Multi-Objective problems*. Genetic and Evolutionary Computation Series. New York: Springer.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on* 6: 182–197.

Demetriou, D., L. See, and J. Stillwell. 2014. Integrating GIS and genetic algorithms for automating land partitioning. In , 9229:922908-922908–9. doi:10.1117/12.2064520.

Eberhart, R., and J. Kennedy. 1995. A new optimizer using particle swarm theory. In , *Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995. MHS '95*, 39–43. doi:10.1109/MHS.1995.494215.

Ehrgott, M. 2005. *Multicriteria optimization*. 2nd ed. Berlin ; New York: Springer.

Ehrgott, M., J. R. Figueira, and S. Greco, ed. 2010. *Trends in Multiple Criteria Decision Analysis*. Vol. 142. International Series in Operations Research & Management Science. Boston, MA: Springer US.

Fister Jr., I., X.-S. Yang, I. Fister, J. Brest, and D. Fister. 2013. A Brief Review of Nature-Inspired Algorithms for Optimization. *arXiv:1307.4186 [cs]*.

Fortin, F.-A., D. Rainville, M.-A. G. Gardner, M. Parizeau, C. Gagné, and others. 2012. DEAP: Evolutionary algorithms made easy. *The Journal of Machine Learning Research* 13: 2171–2175.

Gen, M., R. Cheng, and L. Lin. 2008. *Network Models and Optimization*. Decision Engineering. London: Springer London.

Greene, R., R. Devillers, J. E. Luther, and B. G. Eddy. 2011. GIS-Based Multiple-Criteria Decision Analysis. *Geography Compass* 5: 412–432. doi:10.1111/j.1749-8198.2011.00431.x.

Huang, I. B., J. Keisler, and I. Linkov. 2011. Multi-Criteria decision analysis in environmental sciences: Ten years of applications and trends. *Science of The Total Environment* 409: 3578–3594. doi:10.1016/j.scitotenv.2011.06.022.

Jain, A. K., and R. C. Dubes. 1988. *Algorithms For Clustering Data*.

Jankowski, P. 1995. Integrating geographical information systems and multiple criteria decision-making methods. *International journal of geographical information systems* 9: 251–273. doi:10.1080/02693799508902036.

Jankowski, P., and T. Nyerges. 2003. *Geographic Information Systems for Group Decision Making: Towards a Participatory, Geographic Information Science*. 1st ed. London ; New York: Taylor & Francis e-Library.

Kennedy, J., and R. Eberhart. 1995. Particle swarm optimization. In , *IEEE International Conference on Neural Networks, 1995. Proceedings*, 4:1942–1948 vol.4. doi:10.1109/ICNN.1995.488968.

Kennedy, J., J. F. Kennedy, R. C. Eberhart, and Y. Shi. 2001. *Swarm Intelligence*. Morgan Kaufmann.

Knowles, J., D. Corne, and K. Deb, ed. 2008. *Multiobjective problem solving from nature: from concepts to applications*. Natural Computing Series. Berlin: Springer.

Malczewski, J. 1999. *GIS and Multicriteria Decision Analysis*. John Wiley & Sons.

Malczewski, J. 2000. On the Use of Weighted Linear Combination Method in GIS: Common and Best Practice Approaches. *Transactions in GIS* 4: 5–22. doi:10.1111/1467-9671.00035.

Malczewski, J. 2006. GIS-based multicriteria decision analysis: a survey of the literature. *International Journal of Geographical Information Science* 20: 703–726. doi:10.1080/13658810600661508.

Malczewski, J., and C. Rinner. 2015. *Multicriteria Decision Analysis in Geographic Information Science*. Advances in Geographic Information Science. Berlin, Heidelberg: Springer Berlin Heidelberg.

Mirjalili, S., S. M. Mirjalili, and A. Lewis. 2014. Grey Wolf Optimizer. *Advances in Engineering Software* 69: 46–61. doi:10.1016/j.advengsoft.2013.12.007.

Mirjalili, S., S. Saremi, S. M. Mirjalili, and L. D. S. Coelho. 2016. Multi-Objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Systems with Applications* 47: 106–119. Scopus. doi:10.1016/j.eswa.2015.10.039.

Ngamchai, S., and D. J. Lovell. 2003. Optimal Time Transfer in Bus Transit Route Network Design Using a Genetic Algorithm. *Journal of Transportation Engineering* 129: 510–521. doi:10.1061/(ASCE)0733-947X(2003)129:5(510).

Roy, B. (2005). An overview of MCDA techniques today: paradigms and challenges. In: Figueira, J., Greco, S. and Ehrgott, M. (eds) Multiple criteria decision analysis: state of the art surveys . New York: Springer, pp. 3–24

Saadatseresht, M., A. Mansourian, and M. Taleai. 2009. Evacuation planning using multiobjective evolutionary optimization approach. *European Journal of Operational Research* 198: 305–314. doi:10.1016/j.ejor.2008.07.032.

Sasaki, S., A. J. Comber, H. Suzuki, and C. Brunsdon. 2010. Using genetic algorithms to optimise current and future health planning-the example of ambulance locations. *International journal of health geographics* 9: 1.

Shaygan, M., A. Alimohammadi, A. Mansourian, Z. S. Govara, and S. M. Kalami. 2014. Spatial Multi-Objective Optimization Approach for Land Use Allocation Using NSGA-II. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 7: 906–916. doi:10.1109/JSTARS.2013.2280697.

Son, L. H. 2014. Optimizing Municipal Solid Waste collection using Chaotic Particle Swarm Optimization in GIS based environments: A case study at Danang city, Vietnam. *Expert Systems with Applications* 41: 8062–8074. doi:10.1016/j.eswa.2014.07.020.

Still, G. K. 2014. Standing Crowd Density. December 4.

University of Redlands, and SDS Consortium. 2009. Tools. *Spatial Decision Support Knowledge Portal*. Retrieved 1 May 2016, from http://www.spatial.redlands.edu/sds/

Van Der Walt, S., S. C. Colbert, and G. Varoquaux. 2011. The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering* 13: 22–30. doi:10.1109/MCSE.2011.37.

Wolpert, D. H., and W. G. Macready. 1997. No free lunch theorems for optimization. *Evolutionary Computation, IEEE Transactions on* 1: 67–82.

Yang, X.-S. 2010. *Nature-Inspired Metaheuristic Algorithms*. 2nd ed. Frome, United Kingdom: Luniver Press.

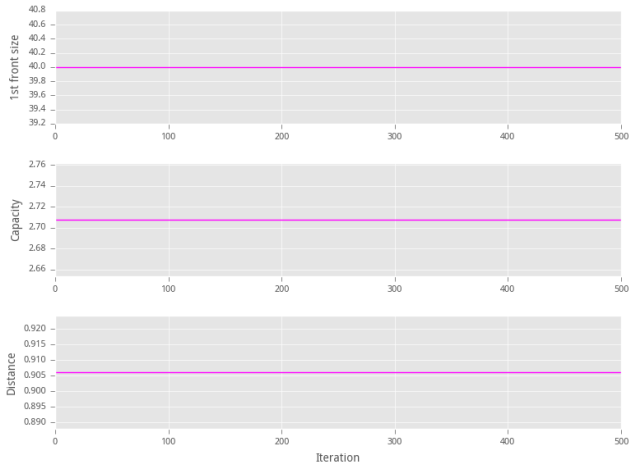
Yang, X.-S. 2014. *Nature-Inspired Optimization Algorithms*. Elsevier.

Zheng, Y.-J., S.-Y. Chen, and H.-F. Ling. 2015. Evolutionary optimization for disaster relief operations: A survey. *Applied Soft Computing* 27: 553–566. doi:10.1016/j.asoc.2014.09.041.

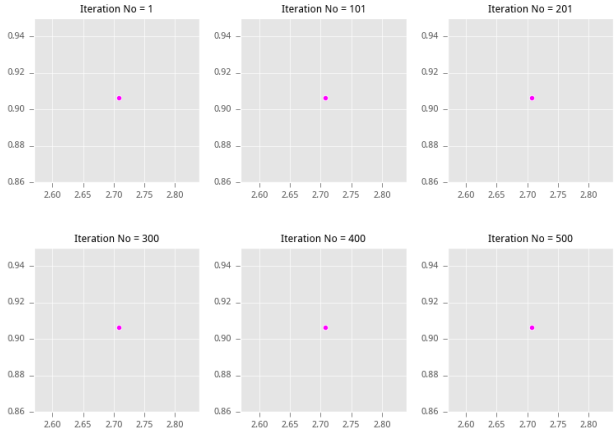
Appendix

In the Results chapter of this report only a summarised set of graphs was presented for specific cases, in order to keep the main text as brief as possible. In this appendix several charts are presented, showing the full output of the optimisation process of each algorithm.

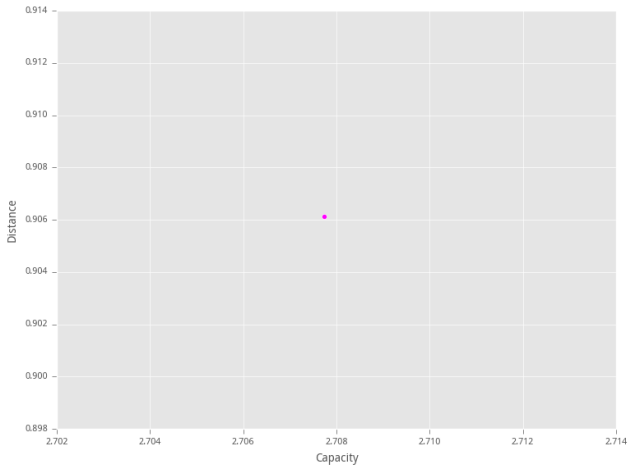
SPSO: Optimisation of Case 0



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for SPSO (Case 0).

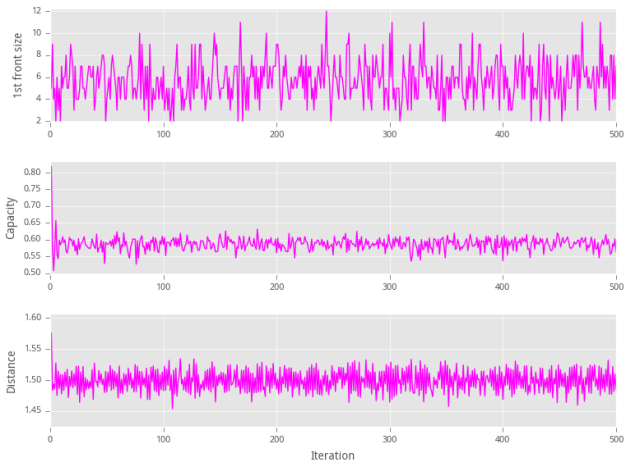


Evolution of the optimisation for the whole swarm in SPSO (Case 0).

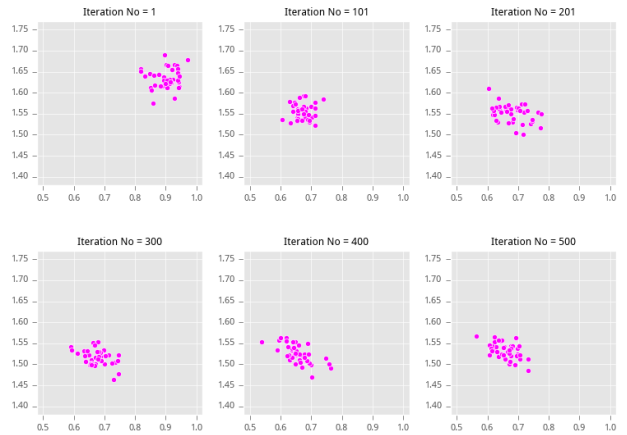


Final 1st Pareto front for SPSO (Case 0).

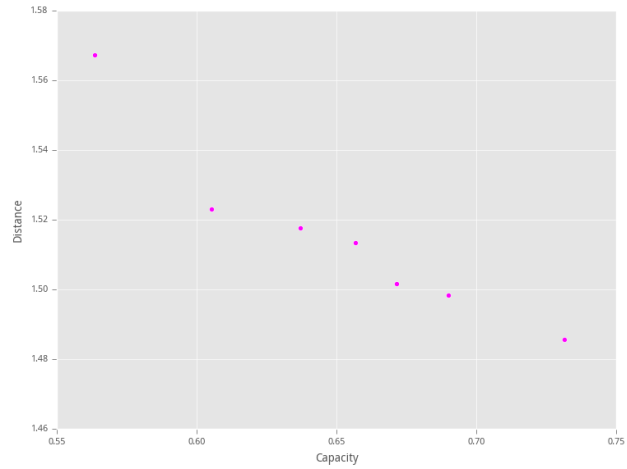
SPSO: Optimisation of Case 1



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for SPSO (Case 1).

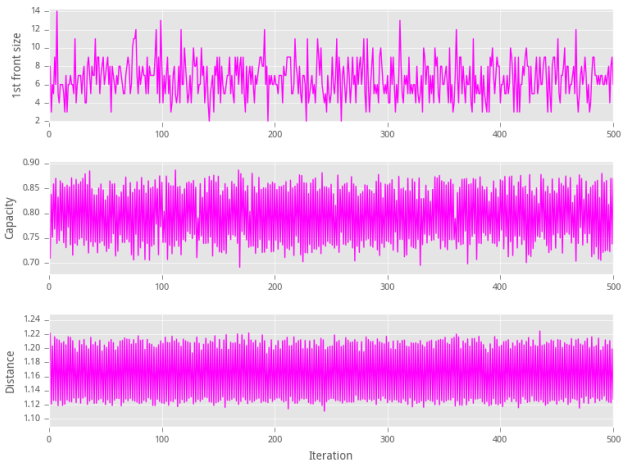


Evolution of the optimisation for the whole swarm in SPSO (Case 1).

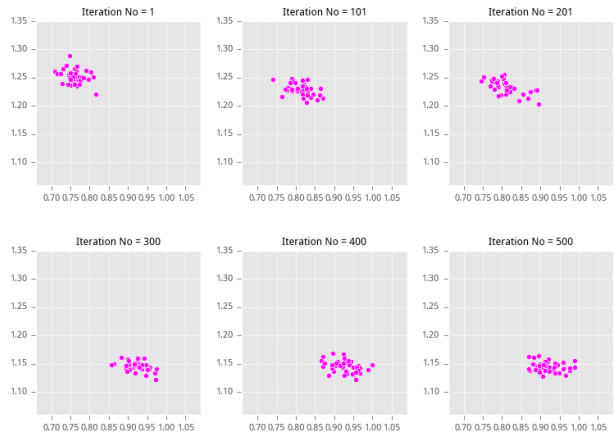


Final 1st Pareto front for SPSO (Case 1).

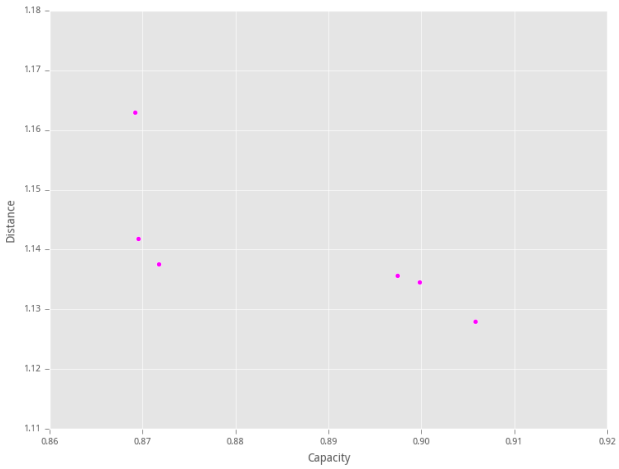
SPSO: Optimisation of Case 2a



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for SPSO (Case 2a).

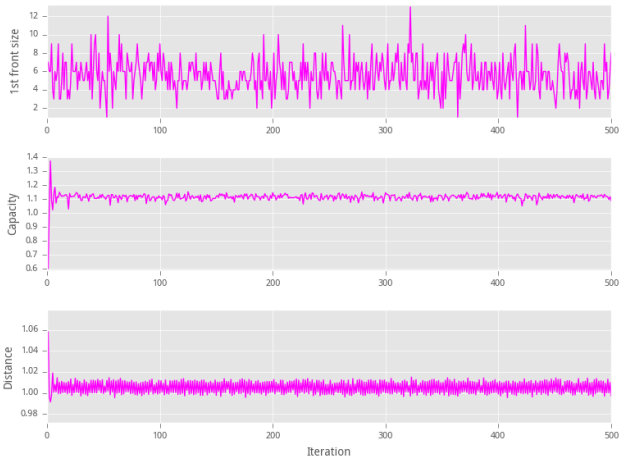


Evolution of the optimisation for the whole swarm in SPSO (Case 2a).

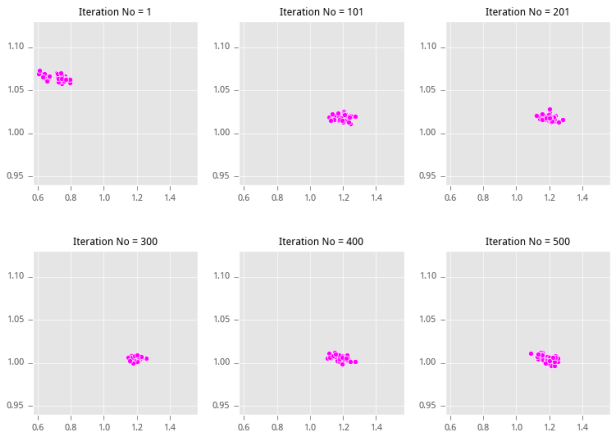


Final 1st Pareto front for SPSO (Case 2a).

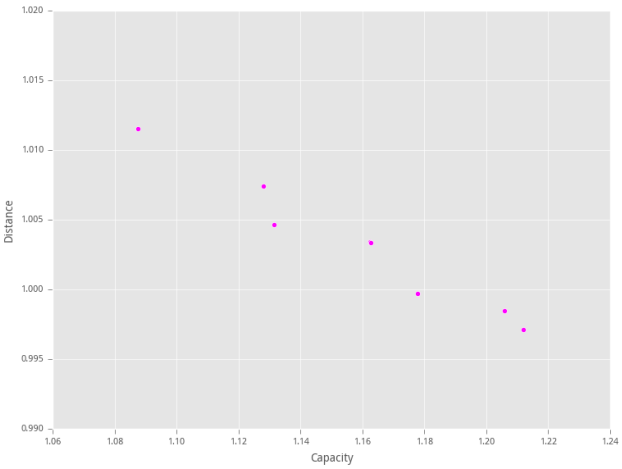
SPSO: Optimisation of Case 2b



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for SPSO (Case 2b).

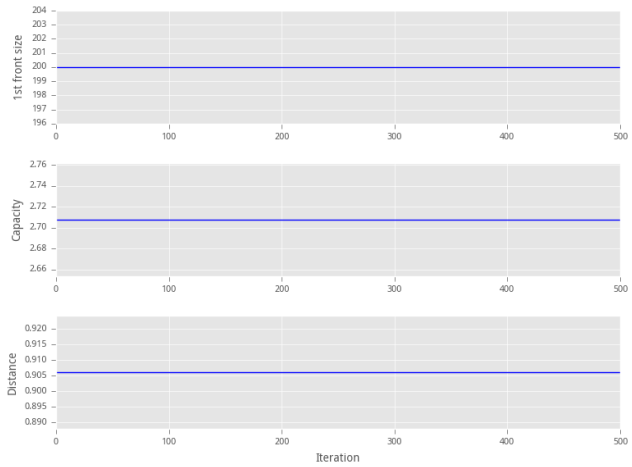


Evolution of the optimisation for the whole swarm in SPSO (Case 2b).

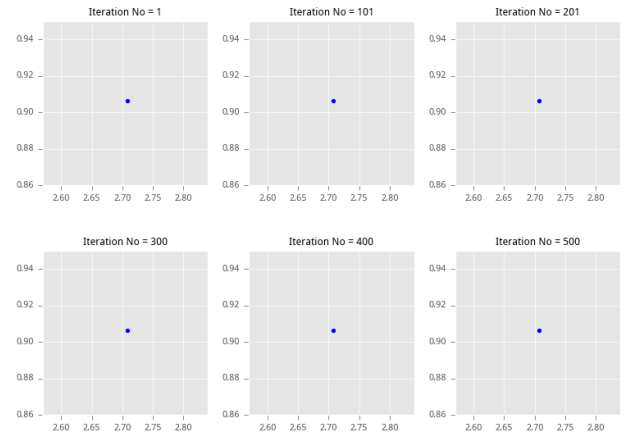


Final 1st Pareto front for SPSO (Case 2b).

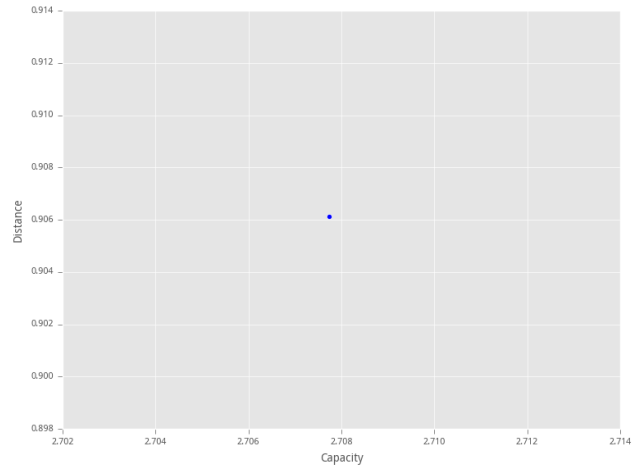
NSGA-II: Optimisation of Case 0



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for NSGA-II (Case 0).

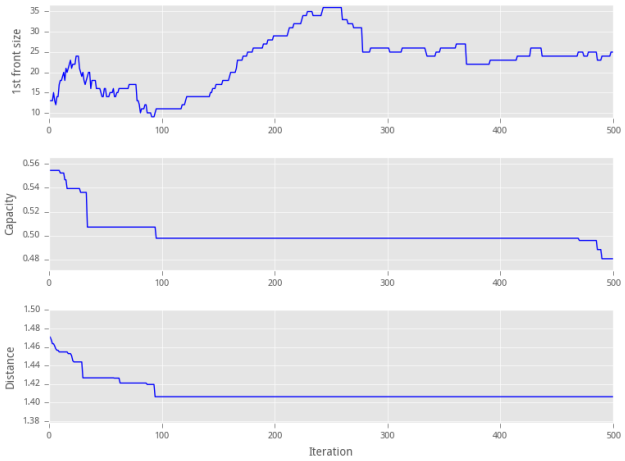


Evolution of the optimisation for the whole population in NSGA-II (Case 0).

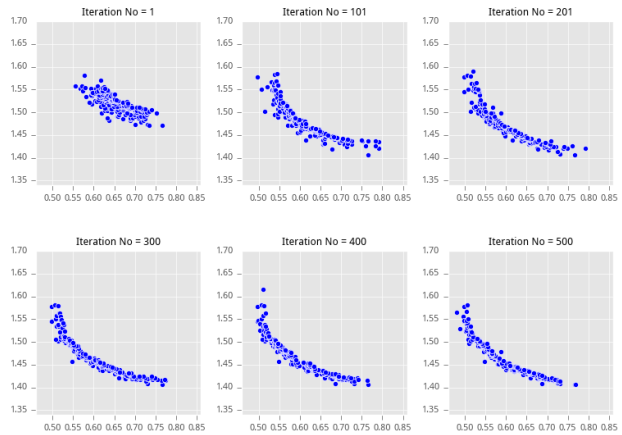


Final 1st Pareto front for NSGA-II (Case 0).

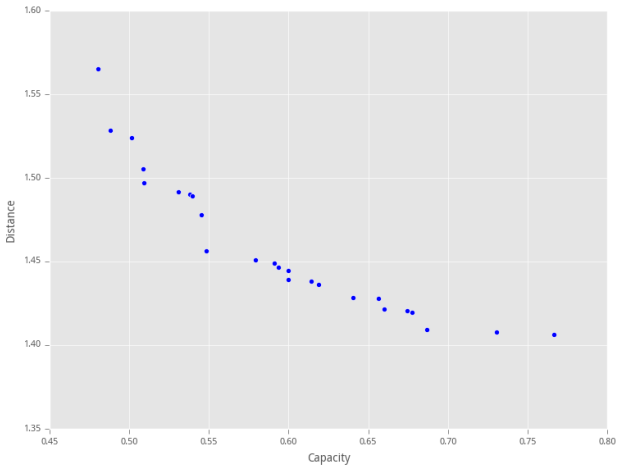
NSGA-II: Optimisation of Case 1



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for NSGA-II (Case 1).

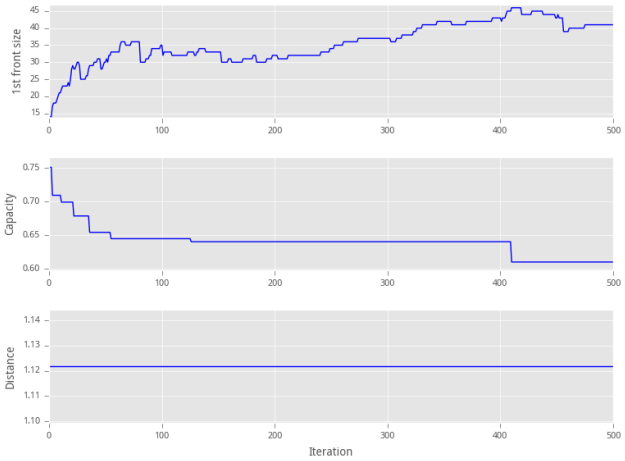


Evolution of the optimisation for the whole population in NSGA-II (Case 1).

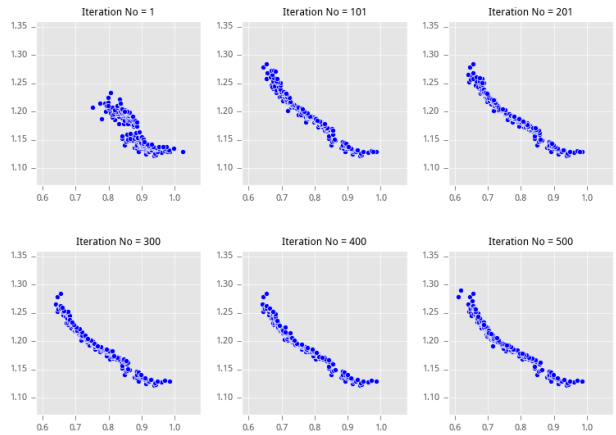


Final 1st Pareto front for NSGA-II (Case 1).

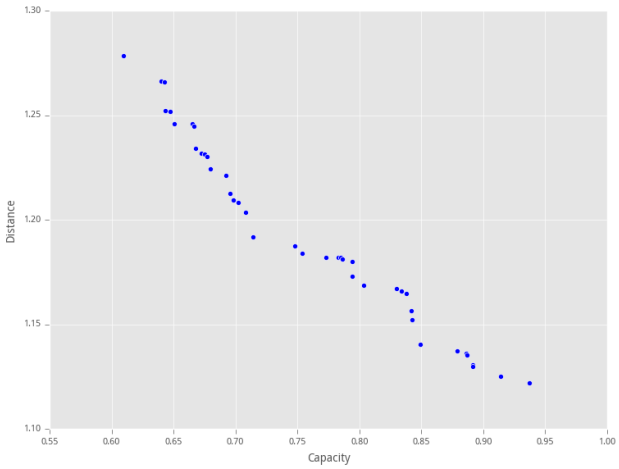
NSGA-II: Optimisation of Case 2a



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for NSGA-II (Case 2a).

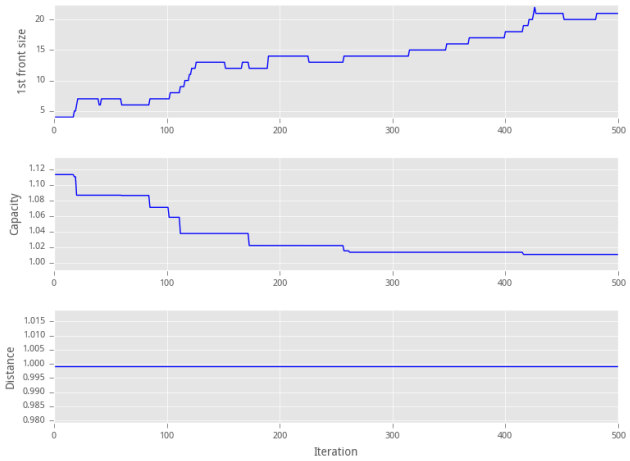


Evolution of the optimisation for the whole population in NSGA-II (Case 2a).

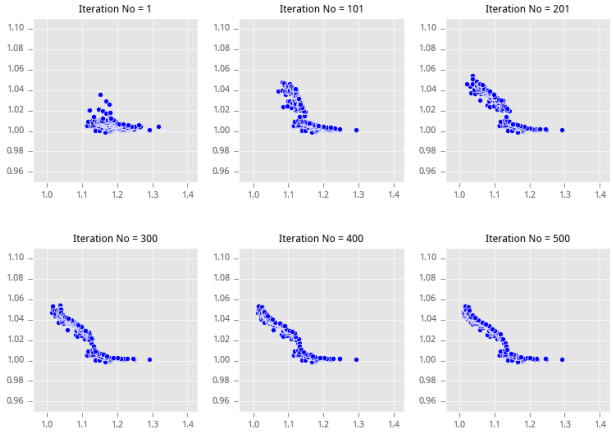


Final 1st Pareto front for NSGA-II (Case 2a).

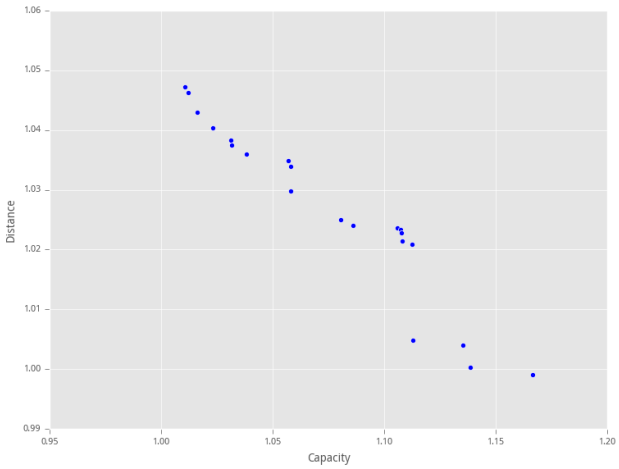
NSGA-II: Optimisation of Case 2b



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for NSGA-II (Case 2b).

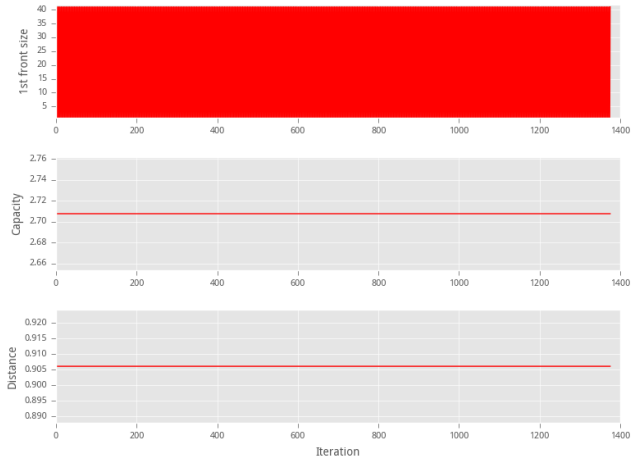


Evolution of the optimisation for the whole population in NSGA-II (Case 2b).

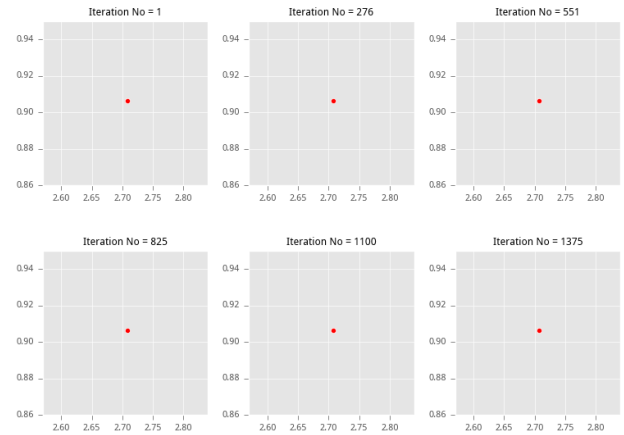


Final 1st Pareto front for NSGA-II (Case 2b).

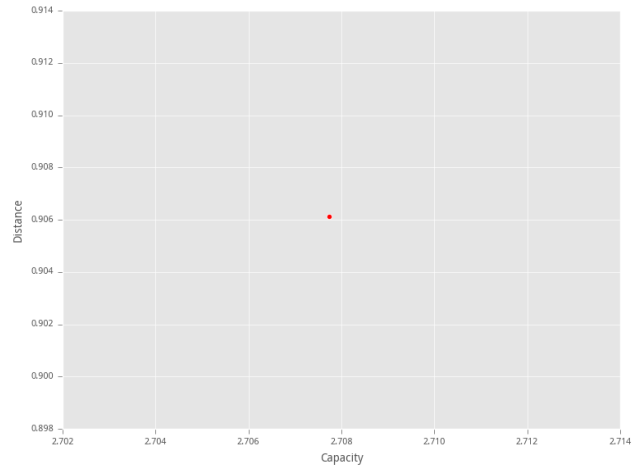
AMOSA: Optimisation of Case 0



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for AMOSA (Case 0).

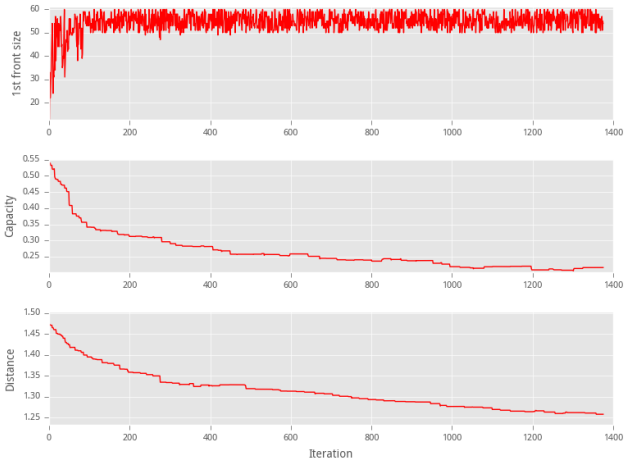


Evolution of the optimisation for the whole archive in AMOSA (Case 0).

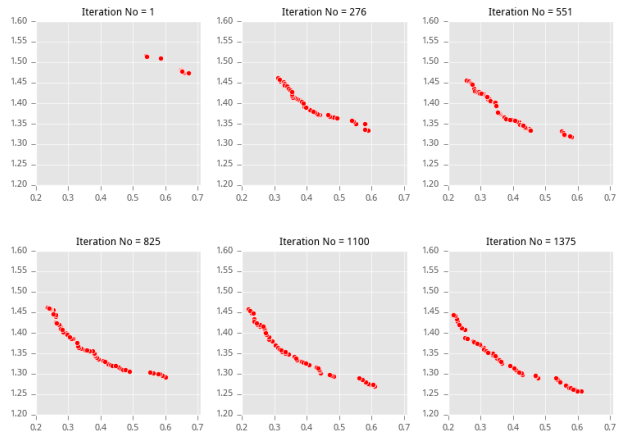


Final 1st Pareto front for AMOSA (Case 0).

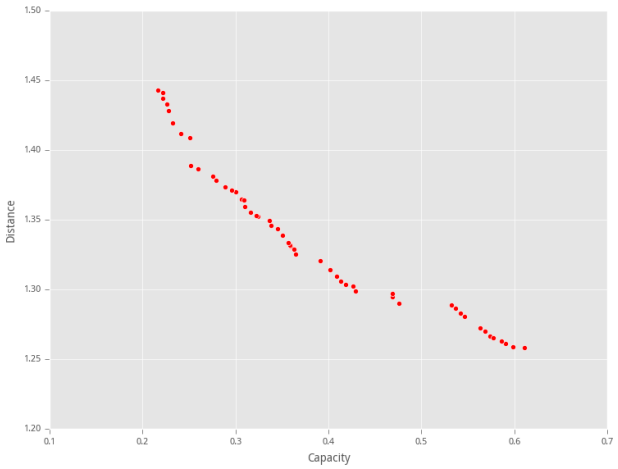
AMOSA: Optimisation of Case 1



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for AMOSA (Case 1).

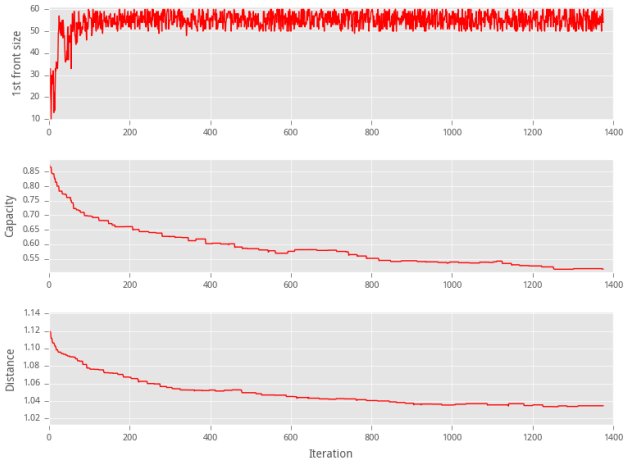


Evolution of the optimisation for the whole archive in AMOSA (Case 1).

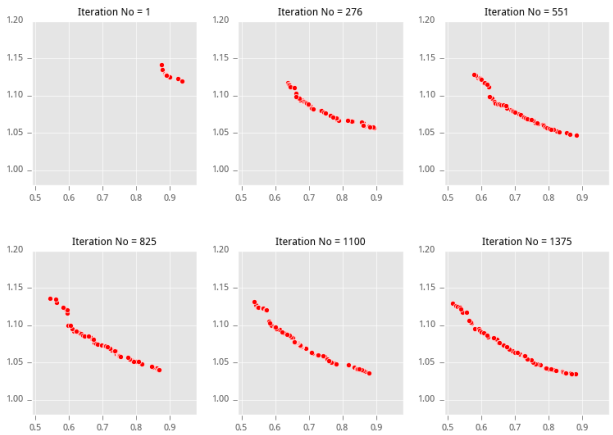


Final 1st Pareto front for AMOSA (Case 1).

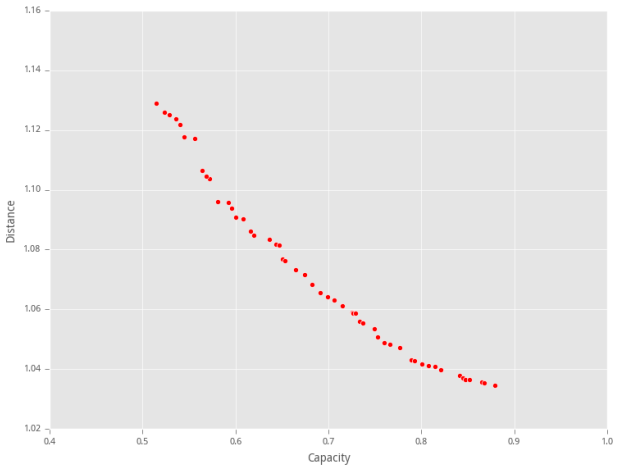
AMOSA: Optimisation of Case 2a



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for AMOSA (Case 2a).

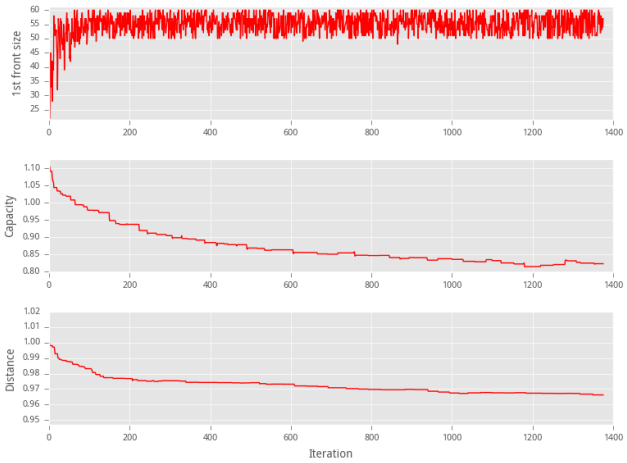


Evolution of the optimisation for the whole archive in AMOSA (Case 2a).

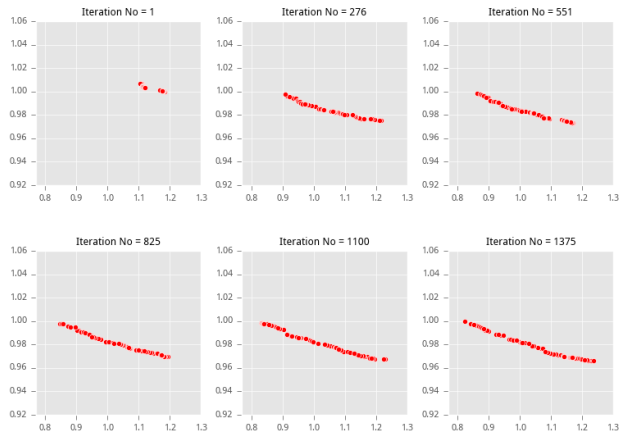


Final 1st Pareto front for AMOSA (Case 2a).

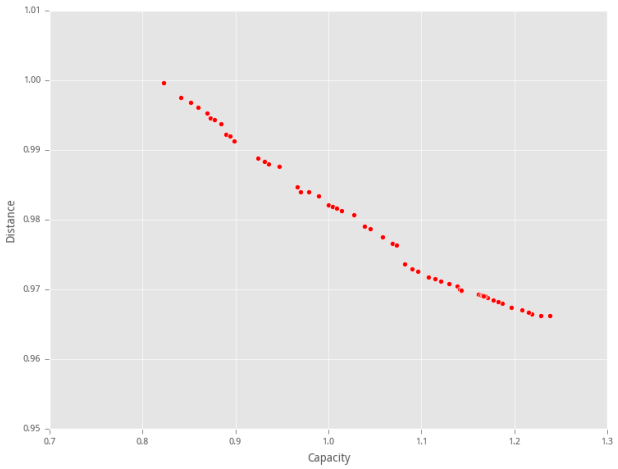
AMOSA: Optimisation of Case 2b



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for AMOSA (Case 2b).

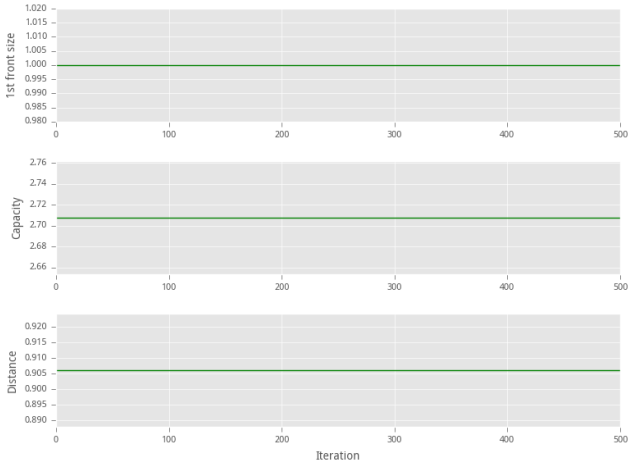


Evolution of the optimisation for the whole archive in AMOSA (Case 2b).

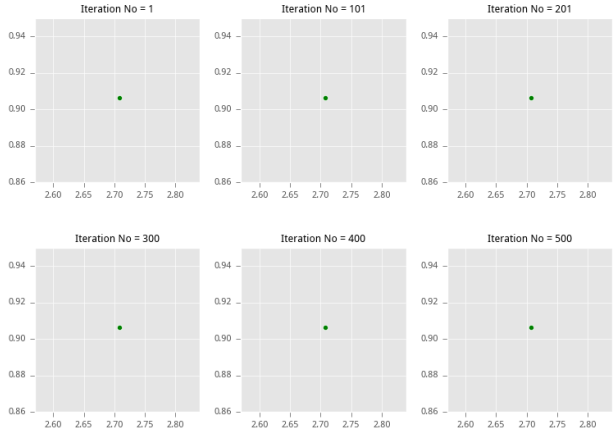


Final 1st Pareto front for AMOSA (Case 2b).

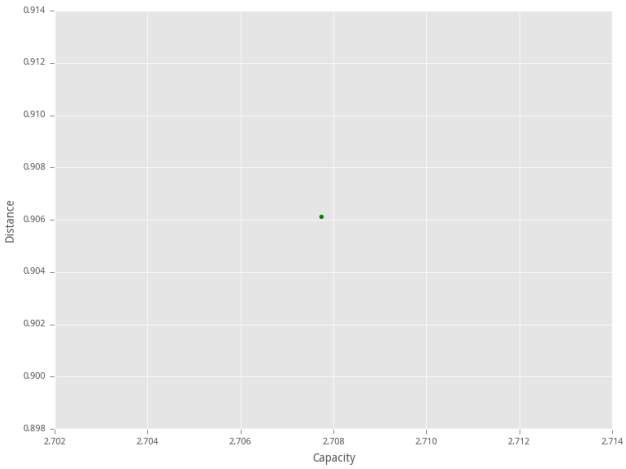
MOGWO: Optimisation of Case 0



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for MOGWO (Case 0).

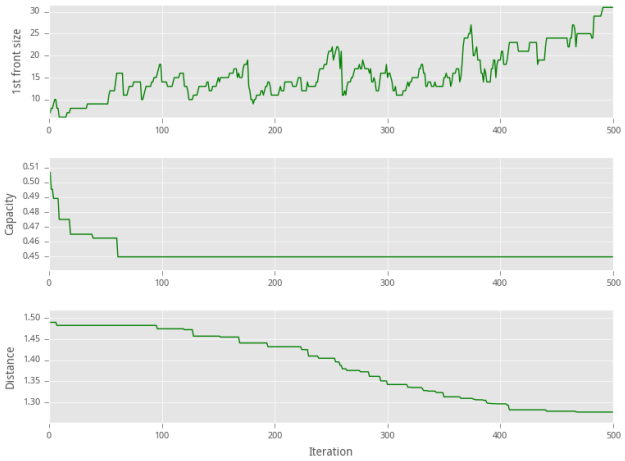


Evolution of the optimisation for the whole pack in MOGWO (Case 0).

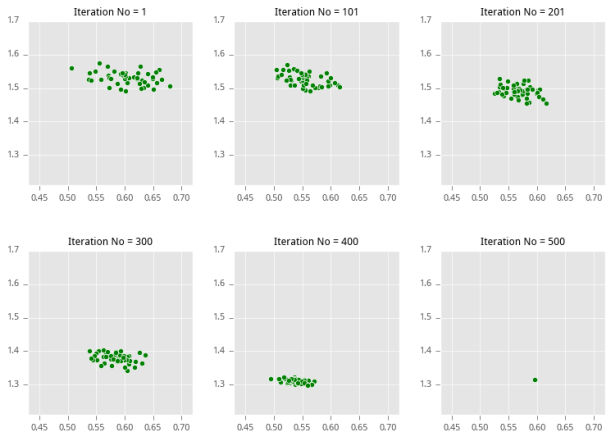


Final 1st Pareto front for MOGWO (Case 0).

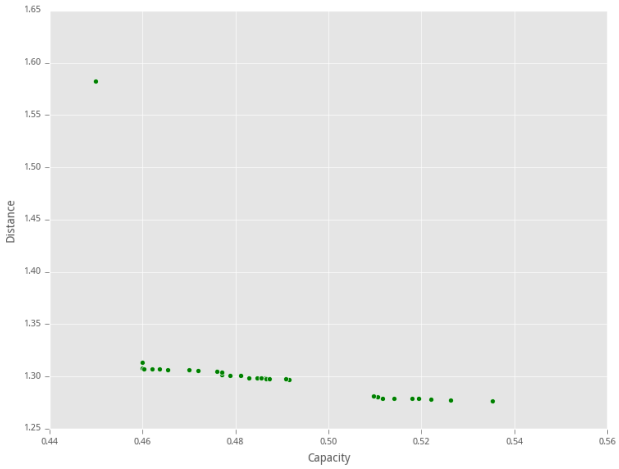
MOGWO: Optimisation of Case 1



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for MOGWO (Case 1).

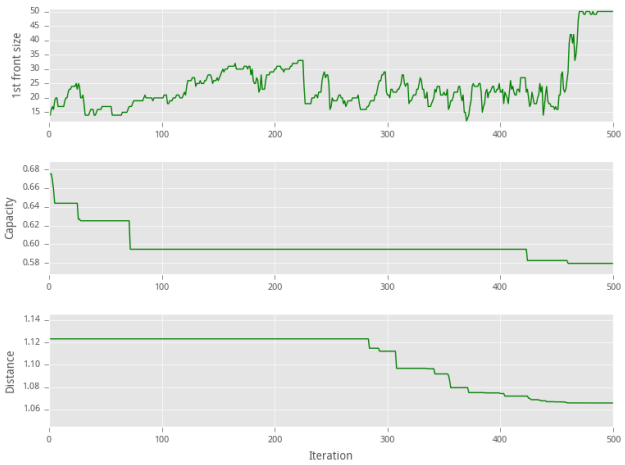


Evolution of the optimisation for the whole pack in MOGWO (Case 1).



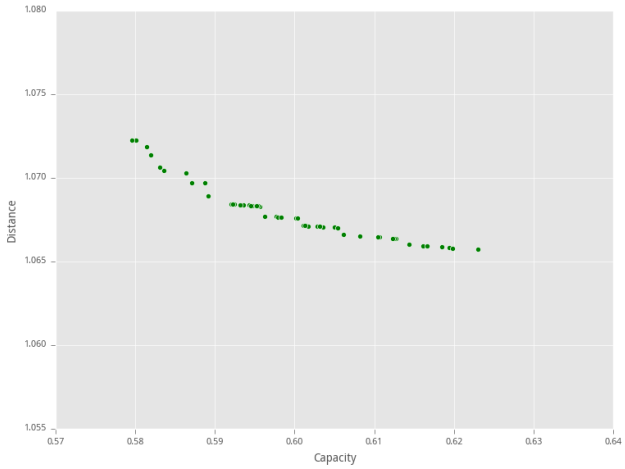
Final 1st Pareto front for MOGWO (Case 1).

MOGWO: Optimisation of Case 2a



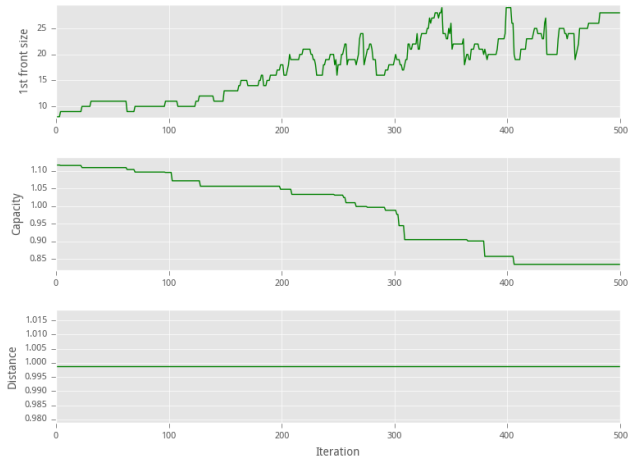
Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for MOGWO (Case 2a).

Evolution of the optimisation for the whole pack in MOGWO (Case 2a).

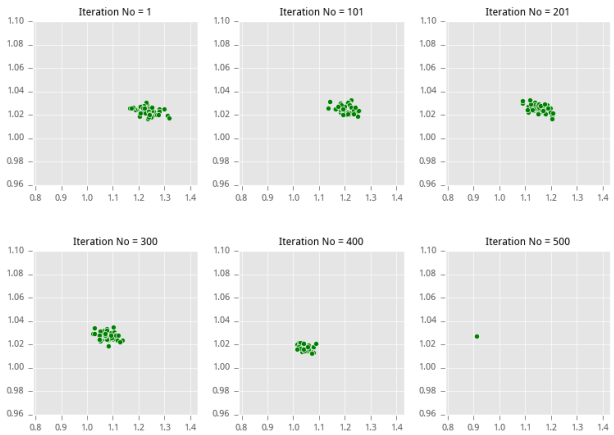


Final 1st Pareto front for MOGWO (Case 2a).

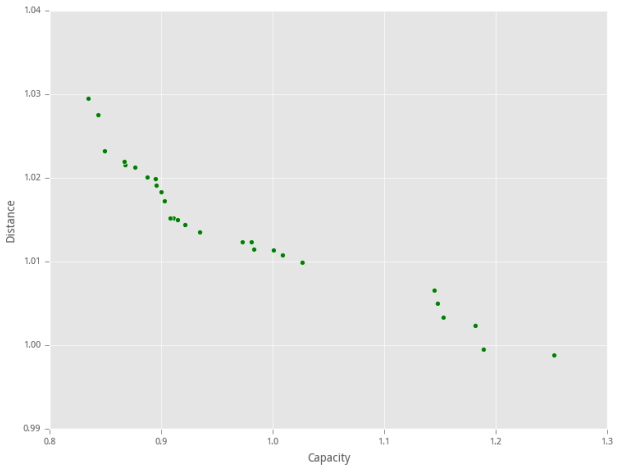
MOGWO: Optimisation of Case 2b



Number of solutions in 1st Pareto front (upper); Optimisation of fcapacity (middle); and Optimisation of fdistance for MOGWO (Case 2b).



Evolution of the optimisation for the whole pack in MOGWO (Case 2b).



Final 1st Pareto front for MOGWO (Case 2b).

Department of Physical Geography and Ecosystem Science, Lund University

Lund University GEM thesis series are master theses written by students of the international master program on Geo-information Science and Earth Observation for Environmental Modelling and Management (GEM). The program is a cooperation of EU universities in Iceland, the Netherlands, Poland, Sweden and UK, as well a partner university in Australia. In this series only master thesis are included of students that performed their project at Lund University. Other theses of this program are available from the ITC, the Netherlands (www.gem-msc.org or www.itc.nl).

The student thesis reports are available at the Geo-Library, Department of Physical Geography and Ecosystem Science, University of Lund, Sölvegatan 12, S-223 62 Lund, Sweden. Report series started 2013. The complete list and electronic versions are also electronic available at the LUP student papers (https://lup.lub.lu.se/student-papers/search/) and through the Geo-library (www.geobib.lu.se).

1 Soheila Youneszadeh Jalili (2013) The effect of land use on land surface temperature in the Netherlands
2 Oskar Löfgren (2013) Using Worldview-2 satellite imagery to detect indicators of high species diversity in grasslands
3 Yang Zhou (2013) Inter-annual memory effects between Soil Moisture and NDVI in the Sahel
4 Efren Lopez Blanco (2014) Assessing the potential of embedding vegetation dynamics into a fire behaviour model: LPJ-GUESS-FARSITE
5 Anna Movsisyan (2014) Climate change impact on water and temperature conditions of forest soils: A case study related to the Swedish forestry sector
6 Liliana Carolina Castillo Villamor (2015) Technical assessment of GeoSUR and comparison with INSPIRE experience in the context of an environmental vulnerability analysis using GeoSUR data
7 Hossein Maazallahi (2015) Switching to the “Golden Age of Natural Gas” with a Focus on Shale Gas Exploitation: A Possible Bridge to Mitigate Climate Change?
8 Mohan Dev Joshi (2015) Impacts of Climate Change on *Abies spectabilis*: An approach integrating Maxent Model (MAXent) and Dynamic Vegetation Model (LPJ-GUESS)
9 Altaaf Mechiche-Alami (2015) Modelling future wheat yields in Spain with LPJ-GUESS and assessing the impacts of earlier planting dates
10 Koffi Unwana Saturday (2015) Petroleum activities, wetland utilization and livelihood changes in Southern Akwa Ibom State, Nigeria: 2003-2015
11 José Ignacio Díaz González (2016) Multi-objective optimisation algorithms for GIS-based multi-criteria decision analysis: an application for evacuation planning