



**Bruno Daniel
Cordeiro Pereira**

**Análise e optimização de sistemas de
abastecimento de água**



**Bruno Daniel
Cordeiro Pereira**

**Análise e optimização de sistemas de
abastecimento de água**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestrado em Engenharia Mecânica, realizada sob orientação científica de António Gil D'Orey de Andrade Campos, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro e de José Paulo de Oliveira Santos, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro.

O júri / The jury

Presidente / President

Doutora Mónica Sandra Abrantes de Oliveira Correia
Professora Auxiliar da Universidade de Aveiro

Vogais / Committee

Doutora Ana Maria Pinto de Moura
Professora Auxiliar da Universidade de Aveiro

Doutor António Gil D'Orey de Andrade Campos
Professor Auxiliar da Universidade de Aveiro (orientador)

Doutor José Paulo de Oliveira Santos
Professor Auxiliar da Departamento de Engenharia Mecânica (co-orientador)

Agradecimentos / Acknowledgements

Gostaria de agradecer, em primeiro lugar, ao meu orientador, Professor Doutor António Gil D'Orey de Andrade Campos e ao meu co-orientador, Professor Doutor José Paulo de Oliveira Santos, pela orientação, apoio e motivação prestados durante a realização desta dissertação.

Gostaria de agradecer de modo especial à minha família, particularmente ao meu pai, Rogério, sem o qual esta caminhada não seria possível, e ao meu irmão, Marcelo, pelo apoio e companheirismo.

Um agradecimento aos amigos que continuam a acreditar em mim e a dar força e apoio.

Por último gostaria de agradecer a todos aqueles que estiveram presentes durante o meu percurso académico e com os quais me tornei a pessoa que sou hoje.

Palavras-chave

Eficiência energética; Métodos de optimização; Simulação hidráulica; Abastecimento de água;

Resumo

Os crescentes consumos de água geram preocupações relacionadas com a sua distribuição. A necessidade de fazer chegar a água a centros populacionais implica elevados custos energéticos e financeiros, pois não existe controlo sobre o bombeamento de água para torres de abastecimento ou reservatórios, a partir das quais se disponibiliza água a uma população, serviços ou indústria. A adaptação do bombeamento de água às tarifas energéticas pode permitir poupanças avultadas a quem executa esse bombeamento. Este trabalho é parte integrante de um projecto de desenvolvimento de um software capaz de, através de modelação hidráulica e ferramentas matemáticas, minimizar os custos de bombeamento e controlar as bombas do sistema de abastecimento de água. Nesta dissertação foram implementados e testados dois algoritmos de optimização para comparar a capacidade de minimização de custos relacionados com o bombeamento da água. Os métodos de optimização seleccionados foram o algoritmo L-BFGS-B (*Limited memory algorithm for bound constrained optimisation*), um método de optimização clássico, e o algoritmo ϵ DE (*epsilon constrained Differential Evolution*), um método metaheurístico. Os algoritmos seleccionados foram testados em funções de teste, tendo o algoritmo ϵ DE obtido bons resultados em todas as funções testadas, enquanto que o algoritmo L-BFGS-B incorreu em dificuldades em funções mais complexas. Os dois algoritmos foram testados em duas redes *benchmark* distintas. Uma rede, denominada Rede Básica, definida apenas pelos elementos essenciais e uma rede malhada denominada rede Walski 489, mais complexa, que inclui duas bombas. Em ambas as redes *benchmark* testadas foram obtidas reduções de custos por ambos os algoritmos implementados. O algoritmo L-BFGS-B provou ser o mais rápido dos algoritmos implementados, enquanto que o algoritmo ϵ DE obteve resultados superiores para a rede mais complexa (rede Walski). Este algoritmo, devido ao facto de testar a violação das restrições em primeiro lugar este tem maior tendência a produzir resultados que não violem essas condições. Foi criada uma interface gráfica que permite o controlo do processo e a apresentação de dados resultantes das optimizações efectuadas.

Keywords

Energy efficiency; optimisation methods; Hydraulic simulation; Water supply

Abstract

Increasing water consumption generates growing concern mainly related to its distribution. The need to get the water to population centres implies high energy consumptions and costs, because there is no control over the pumping of water to supply water towers and reservoirs, from which water is distributed to the population and other services or industry. Suiting the pumping of water having into account energetic tariffs would allow for high financial savings to those who pump water. The present work is part of a current effort to develop a software to achieve the alter, through modulation of a Water Supply System and mathematical tools, minimizing pumping costs via control of the pumps of the so called Water Supply System. In this dissertation were implemented and tested two optimisation algorithms to compare the ability to minimize the costs associated with pumping water. The selected optimisation methods were the L-BFGS-B (Limited memory algorithm for bound constrained optimisation), a classical optimisation algorithm, and the ε DE (epsilon constrained Differential Evolution), a heuristic method. Both algorithms were tested in benchmarked functions, with the ε DE able to provide good results in all functions, while the L-BFGS-B algorithm inferred problems with the more complex functions. Both algorithms were tested in pre-existent benchmarked water networks. One of the networks, denominated Basic Network, simple in nature and with only one pump. The other network, denominated Walski Network, more complex, and with 2 water pumps. Cost reductions were attained with both methods in both benchmarked water networks. The L-BFGS-B algorithm was the fastest of the compared algorithms, while the ε DE algorithm obtained better results than the L-BFGS-B in the Walski Network. The ε DE algorithm is the more assuring to respect the constrains imposed to the networks, as it testes the constraint violations separately. A Graphical User Interface was created to control the process and display the results obtained from optimisation.

Contents

List of Tables	iii
List of Figures	vi
Symbols and Acronyms	vii
I Guidelines	1
1 Introduction	3
1.1 Context	3
1.2 Objectives	4
1.3 Outline of the thesis	4
2 State-of-the-Art Review	5
2.1 Introduction	5
2.2 Water distribution systems	6
2.2.1 Friction losses	7
2.3 Hydraulic simulation	8
2.4 Mathematical optimisation	9
2.4.1 Classical algorithms	10
2.4.2 Modern algorithms	11
2.5 Human Machine Interface	12
2.5.1 Historical review	12
2.5.2 Graphical User Interface Development	12
2.5.3 Characteristics	13
II Methods and Development	15
3 Proposed solution	17
3.1 Optimisation problem formulation	18
3.2 EPANET hydraulic simulator	20
3.2.1 Gradient Method for the solution of hydraulic systems	20
3.3 Selected optimisation algorithms	22
3.3.1 Limited Memory Algorithm for Bound Constrained optimisation	22

3.3.2	ϵ Constrained Differential Evolution	24
3.4	Optimisation variables aggregation	26
3.5	HMI	26
3.6	Developed Graphical User Interface (GUI)	28
III	Results	33
4	Numerical Results	35
4.1	Benchmarks	35
4.1.1	Test Functions	35
4.1.2	Benchmark Results	40
4.2	Basic Network	49
4.2.1	Network Modelling	49
4.2.2	Results Comparison	50
4.3	Walski Network	55
4.3.1	Network Modelling	55
4.3.2	Results Comparison	57
5	Final Remarks	63
5.1	Conclusions	63
5.2	Future work	64
	References	66

List of Tables

2.1	Pipe head loss Formulas for Full Flow	8
4.1	optimisation results of De Jong’s function.	42
4.2	optimisation results of Axis parallel hyper-ellipsoid function.	43
4.3	optimisation results of Rosenbrock’s function.	44
4.4	optimisation results of Easom’s function.	45
4.5	optimisation results of Rastrigin function.	46
4.6	optimisation results of Ackley’s function.	47
4.7	optimisation results of Schwefel’s function.	48
4.8	optimisation results of Michalewicz’s function.	49
4.9	Initial values of energy and cost for the Basic Network	50
4.10	optimisation results of Basic network benchmark.	52
4.11	Initial values of energy and cost for the Walski Network	55
4.12	optimisation results of Walski network benchmark.	58

List of Figures

2.1	Branched network model	6
2.2	Loop network model	7
2.3	Schematic display of the processes involved in the a black-box optimisation.	9
2.4	Multiple local minima and maxima function representation	10
3.1	Schematic display of the processes involved in the proposed solution.	18
3.2	Mock-up of the initial screen of the GUI	27
3.3	Mock-up of the pump control screen of the GUI	27
3.4	Mock-up of the water level screen of the GUI	28
3.5	Mock-up of the final results screen of the GUI	28
3.6	Interface Starting page	29
3.7	Interface Pump control page	30
3.8	Interface Water level page	30
3.9	Interface Estimated savings page	31
4.1	3D representation of the De Jong function	36
4.2	3D representation of the Axis parallel hyper-ellipsoid function	36
4.3	3D representation of the Rosenbrock valley problem	37
4.4	3D representation of Easom's function	38
4.5	3D representation of the Rastrigin Function	39
4.6	3D representation of the Ackley's Function	39
4.7	3D representation of the Schwefel's Function	40
4.8	3D representation of the Michalewicz's Function	41
4.9	De Jong's function optimisation with ϵ DE algorithm	42
4.10	Axis parallel hyper-ellipsoid function optimisation with ϵ DE algorithm	43
4.11	Rosenbrock's valley function optimisation with ϵ DE algorithm	44
4.12	Easom's function optimisation with ϵ DE algorithm	45
4.13	Rastrigin function optimisation with ϵ DE algorithm	46
4.14	Ackley's function optimisation with ϵ DE algorithm	47
4.15	Schwefel's function optimisation with ϵ DE algorithm	48
4.16	Michalewicz's function optimisation with ϵ DE algorithm	49
4.17	Scheme of Basic network.	50
4.18	Consumption pattern associated with Basic Network.	51
4.19	Energy tariff (€) associated with Basic Network.	51
4.20	Characteristic curve of the pump.	51
4.21	Cost function optimisation evolution for Basic Network.	53
4.22	Pump flow and tank level variation after optimisation by ϵ DE	53

4.23	Pump controls after optimisation by the ε DE method.	54
4.24	Pump flow and tank level variation after optimisation by L-BFGS-B	54
4.25	Pump controls after optimisation by the L-BFGS-B method.	55
4.26	Scheme of the Walski network, drawn with EPANET software.	56
4.27	Energy tariff (€) associated with Walski Network.	56
4.28	Consumption patterns associated with Walski Network.	56
4.29	Characteristic curve of the pump.	57
4.30	Cost function optimisation evolution for Walski Network.	59
4.31	Pump flow and tank level variation after optimisation by ε DE	59
4.32	Pump controls after optimisation by the ε DE method	60
4.33	Pump flow and tank level variation after optimisation by L-BFGS-B	60
4.34	Pump controls after optimisation by the L-BFGS-B method	61

Symbols and Acronyms

WSS	Water Supply System
ε DE	ε Constrained Differential Evolution
DE	Differential Evolution
3D	Three Dimensions
GA	Genetic Algorithm
HMI	Human Machine Interface
GUI	Graphical User Interface
CLI	Command Line Interface
NLS	oN-Line System
WYSIWYG	What You See Is What You Get
PC	Personal Computer
IDE	Integrated Development Environment
LGPL	Library General Public Licence
WPF	Windows Presentation Foundation
RDPA	Reduced Dynamic Programming Algorithm
DP	Dynamic Programming
UI	User Interface
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformations
HTML	HyperText Markup Language
XHTML	Extensible HyperText Markup Language
CSS	Cascading Style Sheets
L-BFGS-B	Limited Memory Algorithm for Bound Constrained optimisation

Part I

Guidelines

Chapter 1

Introduction

1.1 Context

Water is the driving force of all nature.

Leonardo da Vinci

Water withdrawals around the world reached an estimated 3900 km³/year [1] each year. As the majority of the population live in urban centres, that generally don't have natural water resources, it becomes necessary to provide water from outer resources. Therefore, water networks are used to conduct water to this high consumption centres. In Portugal, water demands are estimated at 7500 Million m³/year with 5% being destined to urban consumption. However, the estimated costs of water use associated to the urban consumption are of 46% of the total costs [2]. Currently, in water supply systems, it is also necessary to expend energy on a regular basis to accumulate water in the form of potential energy and use it when necessary. The most immediate example is the use of water towers to create pressure on the network or water tanks to supply the population. In the latter example, the water is sent to a higher level by means of pumps. Current systems are taken as imperative to guarantee a minimum level of water for any eventuality. Thus, in the current landscape water is pumped into the towers or supply tanks when the water tank level reaches a minimum value. However, this action does not take into account that the energy cost is dependent on the cycle time. Additionally, the control of pumps is done locally and depends solely on the level sensors. There is no record of running pumps or deposit levels. Costs of these actions can be minimized taking in account the energy cost variation during the day. Energy can be minimized by optimizing the pumping system. When the water supply system contains only one water tank, the task is of small difficulty because the system behaves almost as a linear system and the number of variables to optimize is of reduced number. However, when the system features branches and pumping equipment and tanks multiply, the minimization of energy resources presents itself as a highly complex task. This is due to the large number of variables to optimize, to the non-linear behaviour of the pumps and need of the system to control all organs (pumps, valves, tanks, flow rates in pipes, etc.). The main concern of research in this area is reducing the energy consumption and/or costs through various solutions.

1.2 Objectives

Water supply systems present high energy consumption values due to the pumping systems high energy requirements, necessary to ensure water for the population. On the present situation, the pumping systems are actuated when water levels on water towers reach predefined minimum values. This procedure does not take into account the time of day and the variable cost of energy during the day. The optimisation of the pumping procedure, type of pump, management and logistic relating energy cost, deposit and piping system dimensions could reduce operating costs of water supply systems in a drastic way. This thesis is part of a current effort to develop a software that, through modulation of a Water Supply System and mathematical tools, can predict consumptions, optimize pump controls, reducing energy costs and control the pumps of Water Supply System. The main goal of the present work is to reach cost reductions on water distribution systems through pumping schedule optimisation. The present work aims also to develop a software able to display the results of the optimisation processes to a user.

1.3 Outline of the thesis

The present work is divided in three main parts. The first part, "Guidelines", is divided in two chapters. The first chapter presents an introduction to the theme of work of this project, as well as describe the objectives of said project. The second chapter of the first part is a bibliographical review of themes relevant to this project. This chapter is divided in five sections. In the first is presented a review of previous works on the subject. The second section presents information about Water Supply Systems, while the third section gives information about the hydraulic simulation of said Water Supply System. The fourth section is a review on mathematical optimisation, and the fifth section gives a review of Human Machine Interface development. In the second part of this project, called "Methods and Development", detailed information of the solution used in this project is presented. This part is divided in three chapters, the first one presenting the solution used to model Water Supply System. The second chapter divided in two sections, presents the selected algorithms to use in the Water Supply System optimisation, and the last chapter presents the solution used to develop the Human Machine Interface. The third part of this project is divided in two chapters. The first chapter, divided in three sections, displays the obtained results of the project. The first section presents the results of optimisation of mathematical benchmarks, while the second section presents results for Water Supply System benchmarks and the third section presents the final Human Machine Interface. On the second chapter of this part conclusions from this project are displayed as well as some recommendations for future work.

Chapter 2

State-of-the-Art Review

2.1 Introduction

Water Supply System (WSS) need to ensure the consumption requirements of various sectors of society. These major costs of these systems are usually associated with pumping costs [3], leaving room to improvement on cost efficiency with pump scheduling. To obtain the costs of the variations of pump scheduling, the usage of hydraulic modelling software is advised, as this type of modelling is more complex and able to reproduce the behaviour of WSS more accurately. Water Supply System and hydraulic simulation reviews are addressed in this chapter. The optimisation process of the WSS needs to guarantee flow and pressure conditions in order to satisfy consumers, while reaching pump scheduling controls that minimize cost associated with energy costs that often are associated with time of day. The work of Bagirov et al. [4] introduced the use of pump start/end run times as continuous variables, developing a new algorithm for the solution. The solution is compared to the work of Van Zyl et al. [3] obtaining improvement over the previous paper results. The work of Van Zyl et al.[3] addresses the use of Genetic Algorithm (GA) in WSS. They used successfully an hybrid GA combined with the Hooke and Jeeves Hill-climber Method[5] improving convergence speed and quality of solutions compared to pure GA methods. Both the work of Bagirov et al. [4] and Van Zyl et al.[3] used EPANET software with the same test WSS to evaluate solutions. The work of Wang et al.[6] scheduled the pumping of ground-water taking into account an eco-aware approach to ground-water pumping, scheduling pumping while trying to avoid ground subsidence. Time intervals are represented as real-number arrays instead of binaries, allowing representation of fractions of time intervals. The hydraulic problem used to test the proposed solution is formulated as a discrete-case optimisation problem. The work of Zhuan & Xia [7] analysed the problem of multiple pumps with a Reduced Dynamic Programming Algorithm (RDPA) formulation, reducing computational time comparing to Dynamic Programming (DP) formulation, and being able to reduce costs associated with pumps. To display the data obtained from the optimisation process to the user of the software projected at the overall project exist the need of development of interface between the machine and the user (Human Machine Interface). A review of some characteristics and development tools is provided in this chapter.

2.2 Water distribution systems

Water distribution systems are of great importance as they provide a vital asset to the population. Therefore, it's important to present the characteristics of WSS. Water distribution systems can have branch type layouts, loop layouts or a mix of the two types. In water networks of the branched type the water flows in a single direction, from tank to the last consumption node. A model of this network can be seen in figure 2.1. Looped networks nodes are connected making a grid and are characterized by enabling the water to flow in both directions in pipes between nodes. Water flows depend on the demand in each node. A model of this network can be seen in figure 2.2. Another advantage of this type of network is the lower water velocity in each pipe considering that there are multiple pipes leading to each node[8].

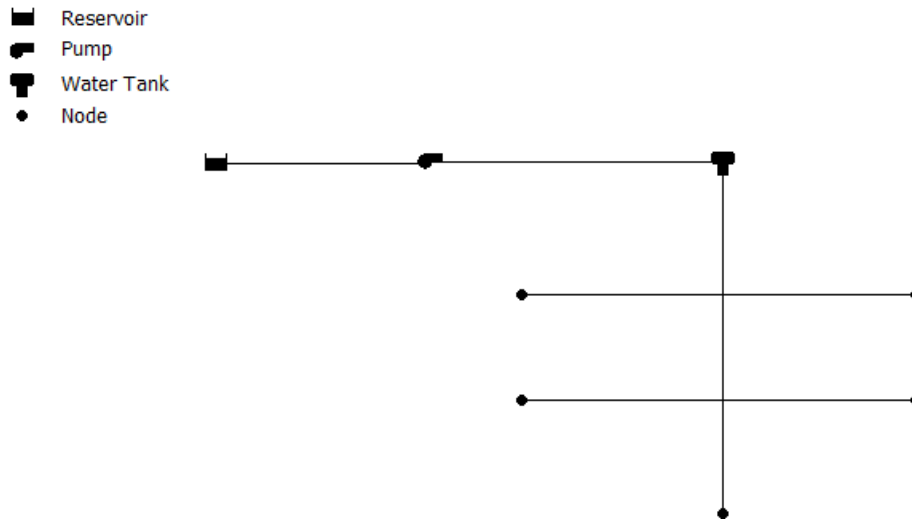


Figure 2.1: Branched network model. In this type of network, the water is distributed throughout the various nodes sequentially.

A water distribution system typically includes:

1. Reservoirs
 - (a) Of variable level, also called tanks. An example of these reservoirs are water towers. These are man made and their water level has significant variations during the time of study.
 - (b) Of fixed level. This category include rivers, lakes or dams. These are usually natural reservoirs, with the exceptions of dams or man made lakes. Their water level does not have significant variations and, consequently, these are considered of fixed level.

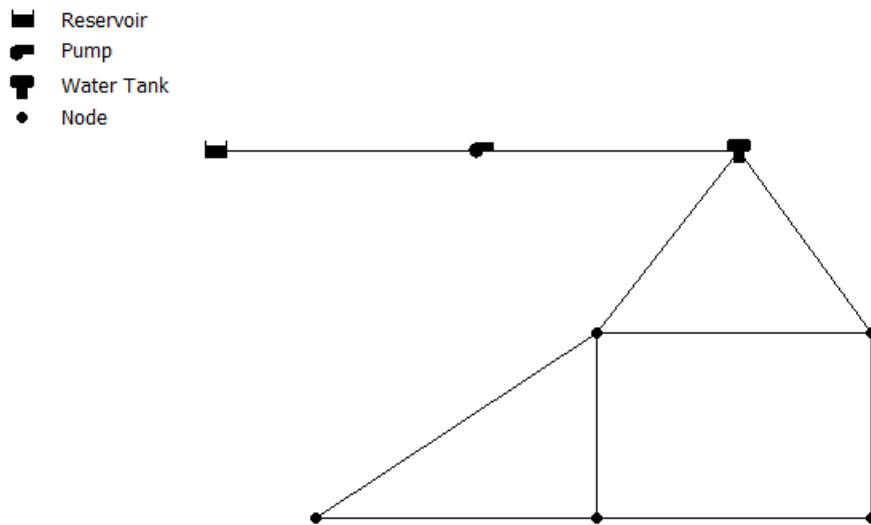


Figure 2.2: Loop network model. In this type of network, the water is distributed throughout the various nodes by a grid of pipes.

2. Pumps. These equipments are used to boost the head at some locations in the network in order to overcome piping head losses and/or to surpass physical elevation differences (like pumping water to an elevated tank). Two types of pumps can be used in water distribution networks, such as:
 - (a) Fixed speed pumps. The motor of the pump remains at a fixed speed regardless of external factors.
 - (b) Variable speed pumps. The motor is connected to a variable speed controller, which controls the rotation of the pump. This type of pumps are more flexible, being used in more applications.
3. Valves. Those allow the water to flow in a given direction, controlling water flow and pressure in a distribution network. Can be used to shut-down entire portions of the networks.
4. Nodes. Junction points, usually connecting two or more pipes. Can be a *dead-end* of a single pipe. Apart from the junction use, nodes can have consumption rates associated or inject inflows (also referred as negative demands).
5. Piping. Join the nodes of the network together and contains water flow.

2.2.1 Friction losses

During the passage of water through the pipes, the friction between water and the pipe walls leads to a head loss. This head loss is calculated using one of the following three

approaches presented:

- Hazen-Williams formula, for head loss in pressure systems. It is the most used formula, however it is only valid for water and was developed for turbulent flow.
- Darcy-Weisbach formula, usable in all liquids and flow regimes.
- Chézy-Manning formula, usable on open conduct problems.

The formulae for the calculation of each approach is presented in table 2.1.

Table 2.1: Pipe head loss Formulas for Full Flow (head loss in meters and flow rate in cubic meters per second) [8].

Formula	Head loss due to friction
Hazen-Williams	$h_L = 10.7C^{-1.852}d^{-4.871}LQ^{1.852}$
Darcy-Weisbach	$h_L = 0.083f(\varepsilon, d, Q)d^{-5}LQ^2$
Chézy-Manning	$h_L = 10.3n^2d^{-5.33}LQ^2$
Notes:	C = Hazen-Williams roughness coefficient ε = Darcy-Weisbach roughness coefficient f = friction factor dependent of ε , d and Q n = Manning roughness coefficient d = pipe diameter in m L = pipe length in m Q = flow rate in m^3/s

2.3 Hydraulic simulation

Simulation software consist of computer based programs that allow modelling, simulation and analysis of steady-state and transient systems, thus allowing to observe an operation without actually performing it. Hydraulic simulators model the system and its components. These are of great importance for water distribution systems management, as they make possible the study of the system previous to its installation. It is possible to ascertain the best option or layout for a piping system, pumping stations or reservoirs easier and quicker, reducing project time and cost and ensure the feasibility of the project. These allow also the improvement of existing systems, providing possible improvements, and are a important tool while studying the behaviour of the system.

The hydraulic model of a simulation is an aggregation of hydraulic components, represented as nodes, which form a network representation of the system being modelled. The physical phenomena are based in macroscopic parameters, which include but are not limited to:

- height of node;
- distance to node;
- size of node.

It's possible to represent complete networks with this approach, enabling a thorough understanding of the hydraulic system.

All of these characteristics prove of great relevance as they endow:

- optimisation of hydraulic networks, when undertaking project design;
- assessment of performance of an existing network, helping to find problems.

2.4 Mathematical optimisation

Nowadays in engineering it is of uttermost importance to consider cost and energy reduction when projecting a process. To improve these reductions, optimisation methods can be applied.

Optimisation processes consist of obtaining the best conditions to operate a process, in order to obtain the best results possible. On the present days, optimisation processes are used in a broad range of applications, such as mechanics, economics and control of industry operations.

Optimisation problems often consist of an attempt to maximize or minimize a mathematical function, called in optimisation theory as objective function. The objective function can depend of one or more variables. In some cases the mathematical function associated with a process is unknown. These cases are usually associated with physical processes, and the mathematical function that represent them are complex. These types of problems are called black-box problems. On this case, reaching the optimal solution becomes harder as the lack of a clear mathematical function blocks access to helpful information. Figure 2.3 displays a schematic of the process followed by black-box optimisation. The optimisation algorithm sends the optimisation variables to the black-box software. After calculation of the objective function and constraints, the black-box software sends the objective function value and constraint values to the optimisation algorithm. This cycle repeats until a defined stopping criteria is reached.

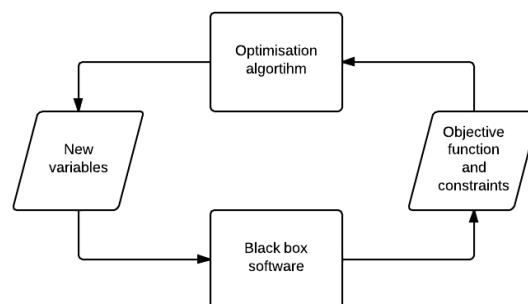


Figure 2.3: Schematic display of the processes involved in the a black-box optimisation.

Objective functions can be linear or non-linear, and can be differentiable or non-differentiable. In the latter, analysis is difficult as differentiable methods cannot be applied.

To reach the best result, variables in the objective function are changed. These are know as optimisation variables. Sometimes constraints are applied to the process, representing

thresholds or requisites that must be verified in the process being optimized. The general optimisation problem can be formulated by:

$$\begin{aligned}
 & \text{minimize} && f(\mathbf{x}) \\
 & \text{subject to} && h(\mathbf{x}) = 0 \\
 & && g(\mathbf{x}) \leq 0 \\
 & && \mathbf{x}_i^{\min} < \mathbf{x}_i < \mathbf{x}_i^{\max},
 \end{aligned} \tag{2.1}$$

where $f(\mathbf{x})$ is the objective function, $i = 1, \dots, n$ is the number of optimisation variables, $h(\mathbf{x})$ are equality constraints and $g(\mathbf{x})$ are inequality constraints, respectively.

2.4.1 Classical algorithms

Classical optimisation methods can use differential calculus, using the gradient of a function to reach the objective. This type of classical algorithms of optimisation can only be used to find the optimal solution of continuous and differentiable functions. Solutions of unknown functions (black-box problems) or of not differentiable functions are harder to solve with these methods.

These methods guarantee that the solution found is exact, but doesn't guarantee that the solution is the best. As example, figure 2.4 is a generic representation of a function which has three local maximums (points A, C and E) and two local minimums (points B and D), being point C a global maximum and point D a global minimum. When using a gradient-based algorithm and using a starting point between A and B, the minimum found will be point B, that is only a local minimum. Additionally, the use of different starting points in multiple runs of the algorithm can lead to different results. Therefore, the use of these methods in non-convex functions is hard to implement and discouraged.

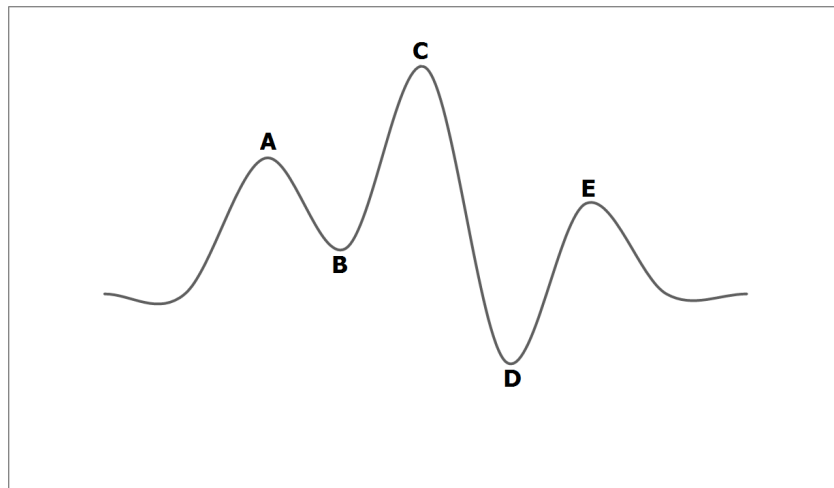


Figure 2.4: Representation of a multiple local minima and maxima function. This type of function presents challenges to classical algorithms.

2.4.2 Modern algorithms

Metaheuristic algorithms are defined as computational methods that use iterations to improve a solution. Although it does not guarantee an optimal solution, the introduction of a random element allows the search for the optimal solution throughout the whole solution spaces. Some metaheuristic methods implement forms of stochastic optimisation. In the example of figure 2.4, for a starting point between point A and point B, on the second iteration the solution tested can be between point C and D (as an example). In the case of a better solution, the previous iteration is discarded. Metaheuristic methods are used to solve complex optimisation problems. These methods are recognized as some of the most practical approaches to complex problems, especially for real-world problems that are combinatorial in nature [9]. These methods are useful in situations where the space of the solution is very large and the approximate solution is not known. Most metaheuristic methods are based in a combination of the random search method and the stochastic hill-climbing method [10]. The random search method strategy is to try a solution from the solution search space using a uniform probability distribution. The strategy used by the stochastic hill-climbing method is randomly selecting a neighbour candidate solution and accepting it only if the result is an improvement [11]. Different types of metaheuristic methods exist, with the search process varying to each one. Stochastic algorithms are based on probabilistic and stochastic processes. Stochastic processes are those whose behaviour is non-deterministic, i.e. randomness is associated with the final output. A deterministic model will always produce the same output from a given starting condition or initial state. The difference between Stochastic Algorithms and other algorithms based on probabilistic and stochastic processes is that Stochastic Algorithms don't have inspiring systems nor metaphorical explanations. These algorithms generate and use random variables.

Evolutionary algorithms are inspired in biological evolution, and uses mechanisms related to it in order to approach a solution. This mechanisms include mutation, reproduction, selection and recombination. Solutions are obtained using the mentioned mechanisms and evaluating a fitness function. Another metaheuristic optimisation method, the physical algorithms are inspired in physical processes, ranging from systems from metallurgy, music, interplay between culture and evolution and complex dynamic systems such as avalanches [11]. Probabilistic Algorithms are those that use probabilist models to model problems or to search problem spaces. These algorithms use the result of a random decision based on probabilistic distribution instead of calculating the best solution. Swarm algorithms are adaptive strategies inspired in collective intelligence. Collective intelligence appears as a the cooperation of multiple individual agents to reach a common goal. Each of the agents is able to sense both itself as its surroundings The aggregation of agents forms a swarm.

Immune algorithms are a part of the Artificial Immune Systems study, which is a class of computational intelligent systems inspired by the process and mechanisms of the biological immune system (primarily mammalian immunology). Neural algorithms make use of artificial neural networks, witch are composed of processing elements, called artificial neurons. Artificial neural networks can have complex global behaviours, as they are affected by the connections between the processing elements of the network and the element parameters. The neural algorithm adapts the weights of connections between elements to produce the desired result.

2.5 Human Machine Interface

A Human Machine Interface (HMI) is what allows interaction between a human and a machine. Their use is widespread from industrial use, as in the screens of machinery, to daily personal use, like the input buttons of a mobile phone. Two types of functions can be present:

- the input from the human user to the machine, to allow adjustments to the machine or to request outputs;
- the display of output from the machine to the user, to, as an example, allow information from the machine to be visible to the user.

2.5.1 Historical review

Human machine interfaces start in history as a necessity of the users to interact with the initial digital computer. At the first times of computer usage, computing power was very limited and expensive. For this machines, interfaces were rudimentary, consisting of punched cards or equivalent as input and line printers as an output. The interaction between user and machine was limited to the system operator console. The first batch systems assigned one job to the entire computer, which could take hours or even days [12]. Command Line Interfaces (CLIs) appeared as an evolution from batch monitors that were connected to the system console. This model interacted with the machine through series of request-response transactions using specialized language to express the requests to the machine. The time of process for this type of interaction dropped significantly from the previous results with the batch system [12]. From the appearance of on-Line System (NLS) with a mouse cursor and multiple windows of hypertext (1968) [13] and the first GUI developed at Xerox PARC, which used windows, icons, and pop-up menus [14], and whose work included the development of the Gypsy, the first bitmap What You See Is What You Get (WYSIWYG).

Apple picked up the work from Xerox PARC and developed Apple Lisa, in 1979, the first personal computer offering a GUI that was directed at individual business users.

With the introduction of 32-bit hardware allowed further development of GUI design. The Microsoft Windows benefited greatly with this development, and introduced their development over their Windows 1.0 (1985) and Windows 2.0 (1987) with the Windows 3.0 (1990)[15]. The mainstream use of computers started in the 1990s created a fast growing market that allowed a high level of competition for commercial development, leading to the appearance of the Windows 95 and the Mac OS, the precursors of the modern GUI present in Personal Computers (PCs). The current development focus is on portable devices and touch-screen interfaces, related with the increasing use of cell phones and tablets seen in the last years. Another area of development is the gesture interface, allowing the user to interact without touching the device.

2.5.2 GUI Development

The GUI development is usually aided by the use of interface builders (or GUI builders), which are software development tools that ease the process of creation. These software tools give the designer a drag and drop WYSIWYG editor, which in turn allow for a quicker visualisation of the GUI being produced. Alternatively, without the GUI builder

the interface must be built by code. This methods does not give visual feedback until the code is executed, impairing design on less experienced programmers.

Some options of software for GUI development include:

- **Visual Studio**
Visual studio is an Integrated Development Environment (IDE) from Microsoft. An IDE is a software that provides tools for software development, normally consisting of source code editors, build automation tools and debuggers. Interpreters and compilers are part of some IDE as well. Visual Studio is used to develop console and GUI applications, as well as Windows Form applications and web sites, applications and services. It can also develop Windows Presentation Foundation (WPF) applications. Visual Studio supports a wide range of programming languages, with C/C++, VB.NET, C# and F# being built-in. It supports XML/XSLT, HTML/XHTML, Javascript and CSS as well. Visual Studio is distributed as a Freeware with the "Express" versions of its components, or as a Trialware on its Professional Editions.
- **GTK+**
GTK+, also know as GIMP [16] toolkit, is a multi-platform toolkit used to create GUIs. The + was added to distinguish between the original version of GTK and the new version [17]. It supports a wide range of programming languages, such as Perl and Python. The GTK+ software is free and is a part of the GNU Project, allowing use by developers,including to develop proprietary software [18].
- **Qt**
Qt is a multi-platform application and User Interface (UI) framework from Digia for developers that uses C++ or QML. It is widely used to develop software applications with GUIs and also to develop non-GUI applications with features like file handling, database access, Extensible Markup Language (XML) parsing, thread management and network support [19]. Qt can be used under open source (Library General Public Licence (LGPL) v2.1) or commercial terms[20].
- **wxWidgets**
wxWidgets is a free and open-source multi-platform C++ library, with bindings for multiple programming languages, such as Python, Perl and Ruby[21]. wxWidgets is currently licensed under the "wxWindows Licence". The wxWindows Licence is essentially the LGPL, with an exception stating that derived works in binary form may be distributed on the user's own terms[22].

2.5.3 Characteristics

The design of a GUI is challenging as it has some important characteristics that it should attend to such as functionality, accessibility, pleasure to use and must be logical to provide quick learning to new users. A poor GUI can undermine a good work, rendering it useless or unsatisfying if its interface is frustrating to the user.

To reach a good interface design, a number of characteristics should be taken into consideration [23]. It should be clear to new users as well as frequent users. If the users can't understand how to work with the interface, it becomes impractical. The interface should be intuitive to the user, allowing the user to naturally and instinctively use the

interface. The interface should be pleasant to the eye and still simple. While challenging, if succeeded it makes the whole experience of the user more enjoyable. The interface should be able to handle mistakes, both from the user and the software. And finally the interface should be a way for the user to accomplish their tasks instead of being a list of possible functions to be used, meaning the interface should be efficient in the goals it wants to achieve.

Part II

Methods and Development

Chapter 3

Proposed solution

The present thesis intends to achieve cost reductions associated with water pumping in Water Supply System. The general optimisation problem associated with this objective can be described as:

$$\begin{aligned} & \text{minimize} && f(\mathbf{x}), \\ & \text{subject to} && h(\mathbf{x}) = 0, \\ & && g(\mathbf{x}) \leq 0, \\ & && \mathbf{x}_i^{\min} < \mathbf{x}_i < \mathbf{x}_i^{\max}, \end{aligned} \tag{3.1}$$

where $f(\mathbf{x})$ is the objective function, $i = 1, \dots, n$ is the number of optimisation variables and $h(\mathbf{x})$ and $g(x)$ are equality and inequality constraints, respectively.

In order to achieve this objective, the proposed solution includes:

- the EPANET software, that produces the hydraulic simulation of the initial case, based on data retrieved by a previous study of the network;
- the use of an optimisation algorithm to improve the pump operation costs, using EPANET to, at each algorithm iteration, produce the new simulation and obtain the new results for operating costs;
- Use a HMI to give the user all informations concerning the changes made to the pump schedule and operation costs, as well as generic informations from the network, such as water level at tanks.

A schematic of the process can be seen in figure 3.1. The process starts with a file containing the network characteristics. This file can be created by the EPANET software, but is a process prior to the optimisation. The data contained in the file is stored in the software responsible for the simulation. The EPANET simulation uses the previous data and runs the WSS simulation. The simulation code produces information sent to the optimisation. This data is the value of the objective function and the constraints information from the latest simulation. After the optimisation process, the new variables produced are sent to the stored data used by the EPANET simulation. This cycle runs until a defined stopping criteria is reached.

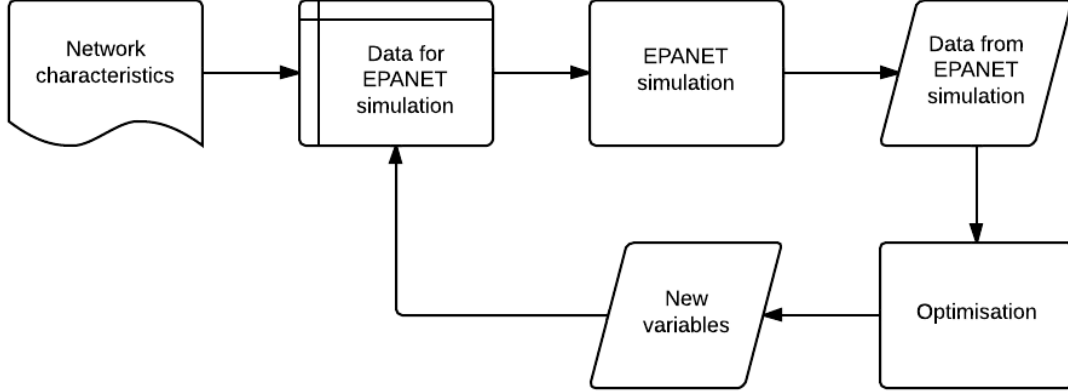


Figure 3.1: Schematic display of the processes involved in the proposed solution.

3.1 Optimisation problem formulation

On the present work the optimisation problem consists in the reduction of costs associated with water pumping in Water Supply System, thus being the objective function. The optimisation variables are the pump controls for a full day. The pumps considered are of variable speed and the considered time step for the controls is of 1 hour. The total number of variables is 48 for each pump, i.e., for each time-step two optimisation variables are associated to each pump, corresponding to the pump speed and the operation time. The objective function is calculated using the software EPANET. As there is no access to the function from EPANET that calculates the costs, the optimisation problem is a black-box problem.

The optimisation problem can be represented by:

$$\begin{aligned}
 & \text{minimize} && f(\mathbf{x}) = \text{Energy cost}, \\
 & \text{subject to} && h(\mathbf{x}) = 0, \\
 & && g(\mathbf{x}) \leq 0, \\
 & && \mathbf{x}_i^{\min} < \mathbf{x}_i < \mathbf{x}_i^{\max},
 \end{aligned} \tag{3.2}$$

where $f(\mathbf{x})$ is the objective function, $i = 1, \dots, n$ is the number of optimisation variables, that include the pump time fraction and the relative velocity of the pump, $h(\mathbf{x})$ are equality constraints and $g(\mathbf{x})$ are inequality constraints. The Energy cost function is calculated as:

$$\text{Energy cost} = \sum_{i=1}^{\text{total steps}} \sum_{j=1}^{\text{total pumps}} [\text{Energy}_{i,j} \times \text{Price}_i] + \text{Fixed Cost}, \tag{3.3}$$

where the Energy for each time step, $i = 1, \dots, \text{total steps}$ and for each pump $j = 1, \dots, \text{total pumps}$, is calculated as:

$$\text{Energy}_{i,j} = P_{i,j} \times t_i, \tag{3.4}$$

with P being the power at the correspondent time step for pump j and t the duration of the pump activation. The power is calculated with:

$$P_{i,j} = \frac{\rho g H_{i,j} Q_{i,j}}{\eta_{i,j}}, \tag{3.5}$$

being ρ the water density, g the standard gravity, H the pump head at the current time step (in meters), Q the flow rate and η is the pump efficiency for pump j . The fixed costs of the energy cost function is calculated with:

$$\text{Fixed cost} = \sum_{j=1}^{\text{total pumps}} P_{j,max} \times \text{Demand charge}, \quad (3.6)$$

with the demand charge being the additional energy charge per maximum kilowatt usage. The pump head is calculated using:

$$H = A - BQ^C, \quad (3.7)$$

where A , B and C are constants related with the pump and Q is the flow rate. With variable speed pumps the head values are shifted according to:

$$\frac{Q_1}{Q_2} = \frac{N_1}{N_2} \quad \frac{H_1}{H_2} = \left(\frac{N_1}{N_2}\right)^2, \quad (3.8)$$

with N_1 and N_2 the standard and the new speed, respectively. The optimisation variables are constrained by

$$0 < \mathbf{x}_i < 1.$$

For the variables of time, the pump time fraction is defined at each time step. For this variable, 0 corresponds to pump working for 0 minutes and 1 to the pump working for 60 minutes. The values between 0 and 1 can be transformed to minutes following a linear equation:

$$\text{time} = \mathbf{x}_i \times 60.$$

For the variables of pump speed, 0 corresponds to pump relative velocity of $\omega = 0.5$ and 1 corresponds to $\omega = 2$. The values between 0 and 1 can be transformed to the relative speed of the pump by the following linear equation:

$$\omega = 0.5 + (\mathbf{x}_i \times 1.5).$$

The optimisation problem is subjected to the following equality constraint:

$$h(\mathbf{x}_j) = L_{j,final} - L_{j,initial} = 0 \quad j = 1, \dots, t, \quad (3.9)$$

with $\mathbf{L}_{initial}$ being the initial water level and \mathbf{L}_{final} the final water level of each tank j .

The optimisation problem is subjected to the following inequality constraint:

$$g1(\mathbf{x}_j) = L_j - L_{j,max} \leq 0 \quad j = 1, \dots, t, \quad (3.10)$$

$$g2(\mathbf{x}_j) = L_j - L_{j,min} \geq 0 \quad j = 1, \dots, t, \quad (3.11)$$

with L_j being the current water level, $L_{j,max}$ the maximum admitted level and $L_{j,min}$ the minimum admitted level for each tank j .

3.2 EPANET hydraulic simulator

The calculation of the objective function of the problem formulated at 3.2 is made by EPANET. EPANET is an hydraulic and water quality simulation software developed by the United States Environment Protection Agency (EPA) and released in 1993. This software allows the simulation of extended period simulations, both static and dynamic. EPANET tracks water flow in pipes, pressure in nodes and height of water in tanks during the simulation period [24]. EPANET can be used as a standalone program or as a library (.dll) to be included in other programs.

EPANET is made of a state-of-the-art hydraulic analysis engine, and is able to[24]:

- model networks with no size restriction;
- model constant or variable speed pumps with an associated curve of function;
- model various types of valves;
- include minor head losses for bends, fittings, etc;
- allow variations of diameter with height in storage tanks;
- associate demand patterns to each individual node;
- calculate pumping energy and cost;
- calculate system operations based on simple tank level or timer controls or base on complex rule based controls;
- calculate friction headloss using the Hazen-Williams, Darcy-Weisbach or Chezy-Manning formulas.

To obtain the solutions for the heads and flows at each time the hydraulic system needs the solving of the equation for the conservation of flow at each junction and the headloss across each link of the water network. These equations gives the hydraulic balance of the network at a given time. EPANET hydraulic simulation model employs a gradient method in order to solve the non-linear equations involved in the hydraulic balance.

3.2.1 Gradient Method for the solution of hydraulic systems

EPANET uses an approach from Todini and Pilati(1988) [25] to solve the equations that characterize the hydraulic balance of the network. This approach is presented next.

The flow-headloss relation in a defined pipe between the nodes i and j is given by:

$$H_i - H_j = h_{ij} = rQ_{ij}^n + mQ_{ij}^2, \quad (3.12)$$

where H is the nodal head, h is the headloss, r is the resistance coefficient, Q is the flow rate, n is the flow exponent and m is the minor loss coefficient. The value of the resistance coefficient is dependant of the friction headloss formula being used. The headloss for pumps can be represented by

$$h_{ij} = -\omega^2(h_0 - r(\frac{Q_{ij}}{\omega})^n), \quad (3.13)$$

in which h_0 is the head of shut-off for the pump, ω is a relative speed setting, and r and n are the pump curve coefficients.

To attain the hydraulic balance, another set of equations must be satisfied. These are the flow continuity equations for all nodes:

$$\sum_j Q_{ij} - D_i = 0 \quad \text{for } i = 1, \dots, N, \quad (3.14)$$

in which D_i is the flow demand in the node i . By convention, the flow into a node is positive. The objective of the balance is to find heads H_i and flows Q_{ij} that satisfy equations 3.12 and 3.14.

The gradient method starts with a first estimate of flows in pipes that may not satisfy flow continuity. From each iteration the new nodal heads are obtained solving the matrix equation:

$$\mathbf{A}\mathbf{H} = \mathbf{F}, \quad (3.15)$$

where \mathbf{A} is an $(N \times N)$ Jacobian matrix, \mathbf{H} is an $(N \times 1)$ vector of unknown nodal heads and \mathbf{F} is an $(N \times 1)$ vector of right hand side terms.

The diagonal elements of the \mathbf{A} matrix are given by:

$$A_{ii} = \sum_j p_{ij}, \quad (3.16)$$

and the non-zero off-diagonal elements are given by:

$$A_{ij} = -p_{ij}, \quad (3.17)$$

where p_{ij} is the inverse derivative of the headloss in the link between the respective nodes with respect to flow. For pumps, p_{ij} is given by

$$p_{ij} = \frac{1}{n\omega^2 r \left(\frac{Q_{ij}}{\omega}\right)^{n-1}}, \quad (3.18)$$

while for pipes p_{ij} is given by

$$p_{ij} = \frac{1}{nr|Q_{ij}|^{n-1} + 2m|Q_{ij}|}. \quad (3.19)$$

The \mathbf{F} vector consists of net flow imbalances at the node added to a flow correction factor:

$$F_i = \left(\sum_j Q_{ij} - D_i \right) + \sum_j y_{ij} + \sum_f p_{if} H_f, \quad (3.20)$$

in which the last term of the equation applies to any links that connect node i to a fixed grade node f . The flow correction factor y_{ij} for pipes is given by:

$$y_{ij} = p_{ij} \left(r|Q_{ij}|^n + m|Q_{ij}|^2 \right) \text{sgn}(Q_{ij}), \quad (3.21)$$

and for pumps it is given by:

$$y_{ij} = -p_{ij}\omega^2 \left(h_0 - r\left(\frac{Q_{ij}}{\omega}\right)^n \right), \quad (3.22)$$

where $\text{sgn}(Q_{ij})$ is 1 when Q_{ij} is positive and -1 otherwise. Q_{ij} is always positive for pumps, hence this term is omitted in the equation of pumps. After the calculation of new heads by solving equation 3.15 the new flows are calculated using:

$$Q_{ij} = Q_{ij} - (y_{ij} - p_{ij}(H_i - H_j)). \quad (3.23)$$

The results are tested against a pre-determined tolerance of the sum of absolute flow relative to the total flow in all links. If the tolerance is not respected, equation 3.15 and 3.23 are solved again.

The implementation of the method in EPANET follows some essential steps, namely:

1. The linear system of equations 3.15 is solved with use of a sparse matrix method based on node re-ordering;
2. At the first iteration, flow in a pipe is assumed to be equal to the flow corresponding to a velocity of 1 ft/sec (30,48 cm/sec) and the flow in pumps is equal to the design flow specific of the pump;
3. The resistance coefficient for a pipe (r) is calculated based on one of three different approaches, concretely:
 - Hazen-Williams formula.
 - Darcy-Weisbach formula.
 - Chézy-Manning formula.

The equations for each formulation are present in table 2.1, previously presented in section 2.2.1.

4. The minor loss coefficient defined in order of velocity head K is converted to a flow-based coefficient with the following equation:

$$m = \frac{0.02517K}{d^4}. \quad (3.24)$$

3.3 Selected optimisation algorithms

To solve the optimisation problem formulated at 3.2 two different algorithms are proposed. The Limited Memory Algorithm for Bound Constrained optimisation (L-BFGS-B), a classical algorithm and the ε Constrained Differential Evolution (ε DE), a modern algorithm. Both algorithms are presented in the next two sections.

3.3.1 Limited Memory Algorithm for Bound Constrained optimisation

The L-BFGS-B is a limited memory quasi-Newton algorithm, used to solve large non-linear optimisation problems, in which there are simple bounds on the problem variables [26]. The problem on this algorithm is formulated as

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) \\ &\text{subject to} && \mathbf{l} < \mathbf{x} < \mathbf{u}, \end{aligned} \quad (3.25)$$

where $f : \mathfrak{R}^n \rightarrow \mathfrak{R}$ is a non-linear function with an available gradient function \mathbf{g} , in which the vectors \mathbf{l} and \mathbf{u} represent the lower and higher bounds of the variables, respectively, and the number of variables, n , is assumed to be large. The gradient function \mathbf{g} is continuous.

The formulated optimisation problem is subjected to the following equality constraint:

$$h(x_j) = L_{j,final} - L_{j,initial} = 0 \quad j = 1, \dots, t, \quad (3.26)$$

with $L_{initial}$ being the initial water level and L_{final} the final water level of each tank j .

The optimisation problem is subjected to the following inequality constraint:

$$g1(x_j) = L_j - L_{j,max} \leq 0 \quad j = 1, \dots, t, \quad (3.27)$$

$$g2(x_j) = L_j - L_{j,min} \geq 0 \quad j = 1, \dots, t, \quad (3.28)$$

with L_j being the current water level, $L_{j,max}$ the maximum admitted level and $L_{j,min}$ the minimum admitted level for each tank j .

The mathematical description of the algorithm was described by its authors, Richard H. Byrd et al. in 1994 [26]. For this algorithm, the gradient function \mathbf{g} is calculated using a finite difference method called the forward difference, which is represented by:

$$\Delta_h [f](\mathbf{x}) = f(\mathbf{x} + h) - f(\mathbf{x}). \quad (3.29)$$

The derivative of function f at \mathbf{x} is given by:

$$f'(\mathbf{x}) = \lim_{h \rightarrow +\infty} \frac{f(\mathbf{x} + h) - f(\mathbf{x})}{h}. \quad (3.30)$$

For small h and $h \neq 0$ the forward difference method approximates the derivative of $f(\mathbf{x})$ as:

$$f'(\mathbf{x}) \approx \frac{f(\mathbf{x} + h) - f(\mathbf{x})}{h} = \frac{\Delta_h [f](\mathbf{x})}{h}. \quad (3.31)$$

The constraints from the formulated optimisation problem are added to the algorithm using the exterior penalties method, which penalises the objective function using:

$$F = f + r_h \sum_{k=1}^l (h_k(X))^2 + r_g \sum_{j=1}^m (\max\{0, g_j(X)\})^2, \quad (3.32)$$

where F is the objective function after penalization, f is the objective function prior to penalization, r_h is the coefficient for the equality constraints and r_g is the coefficient for the inequality constraints.

For the implementation of this optimisation algorithm a C++ code, containing around 2000 lines was developed [26]. Besides the adaptation to the type of problem intended to optimize in this work, one of the main differences introduced in the code was the implementation of a constraint handling method based on the exterior penalties method, referred above. Further inclusions in this code include the gradient calculation for the objective function, based on the finite difference method of the forward differences. These inclusions, made in C++, allowed the use of this algorithm in the problem studied.

3.3.2 ε Constrained Differential Evolution

Being a part of the Stochastic Direct Search methods, Differential Evolution (DE) is from a field of Evolutionary Computation, being related with methods such as Genetic Algorithms, Evolutionary Programming and Evolution Strategies. DE was designed for non-linear, non-differentiable continuous function optimisation [11].

DE algorithms have a population of candidate solutions, which are used through iterations of recombination, evaluation and selection to achieve the optimal result.

The recombination of candidate solutions is based in the weighed difference between two random selected candidates (vectors \mathbf{b} and \mathbf{c}) added to a third candidate solution (vector \mathbf{a}). The resulting candidate is mutated with a crossing vector \mathbf{i} . After this process, the created candidate solutions are tested against the progenitor candidates. If better, the child candidate replaces the father in the population of candidate solutions. With this method, while the population of candidates is spread out the variations made at each iteration will be high. As the solution converges, the changes become smaller as the distance between the candidates selected for subtraction (\mathbf{b} and \mathbf{c}) are smaller. To note as well is the fact that the selection in this method is made after the recombination iterations making this a survival selection instead of having parent selection.

A simple implementation of a DE is show below in algorithm 1. In the presented case the population is treated as a vector to improve clearness of the code.

The necessity of guaranteeing water level constraints in the WSS problems leads to addition of constraint manage to DE algorithm. The algorithm proposed by Takahama and Sakai [27], which is used in this work, addresses this problem adding the ε constrained method to the standard DE algorithm. The ε constrained method uses constraint violations, $\phi(x)$ which is given by [27]

$$\phi(x) = \max \{ \max \{0, g_j(x)\}, \max |h_j(x)| \}, \quad (3.33)$$

$$\phi(x) = \sum_j \|\max \{0, g_j(x)\}\|^p + \sum_j \|h_j(x)\|^p, \quad (3.34)$$

with p being a positive number. The ε level comparison defines the order relation of a pair of objective functions, value and constraint violation ($f(x), \phi(x)$). The ε level comparison defines the order of precedence of $\phi(x)$ over $f(x)$, because the feasibility of x is more important than the minimization of $f(x)$. For f_1, f_2 and ϕ_1, ϕ_2 being the function values and constraint violations at the point x_1, x_2 the ε level comparison for any $\varepsilon \geq 0$ the $<_\varepsilon$ and \leq_ε between (f_1, ϕ_1) and (f_2, ϕ_2) are defined as:

$$(f_1, \phi_1) <_\varepsilon (f_2, \phi_2) \Leftrightarrow \begin{cases} f_1 < f_2, & \text{if } \phi_1, \phi_2 \leq \varepsilon, \\ f_1 < f_2, & \text{if } \phi_1 = \phi_2, \\ \phi_1 < \phi_2, & \text{otherwise,} \end{cases} \quad (3.35)$$

$$(f_1, \phi_1) \leq_\varepsilon (f_2, \phi_2) \Leftrightarrow \begin{cases} f_1 \leq f_2, & \text{if } \phi_1, \phi_2 \leq \varepsilon, \\ f_1 \leq f_2, & \text{if } \phi_1 = \phi_2, \\ \phi_1 < \phi_2, & \text{otherwise.} \end{cases} \quad (3.36)$$

For the case of $\varepsilon = \inf$ the comparison is equivalent to ordinary comparisons. For the case of $\varepsilon = 0$ the comparison orders the constraint violation $\phi(x)$ precedes de function value $f(x)$.

Algorithm 1 DE pseudo-code

```

1:  $\alpha \leftarrow$  mutation rate  $\triangleright$  Commonly between 0.5 and 1.0, higher is more explorative
2:  $popsiz$   $\leftarrow$  desired population size
3:  $P \leftarrow \langle \rangle$   $\triangleright$  Empty population(it's convenient here to treat it as a vector), of length  $popsiz$ 
4:  $Q1 \leftarrow \square$   $\triangleright$  The parents. Each parent  $Q_i$  was responsible for creating the child  $P_i$ 
5: for  $i$  from 1 to  $popsiz$  do
6:    $P_i \leftarrow$  New random individual
7: end for
8:  $Best \leftarrow \square$ 
9: repeat
10:  for each individual  $P_i \in P$  do
11:    AssessFitness( $P_i$ )
12:    if  $Q \neq \square$  and  $Fitness(Q_i) > Fitness(P_i)$  then
13:       $P_i \leftarrow Q_i$   $\triangleright$  Retain the parent, throw away the kid
14:    end if
15:    if  $Best = \square$  or  $Fitness(P_i) > Fitness(Best)$  then
16:       $Best \leftarrow P_i$ 
17:    end if
18:     $Q \leftarrow P$ 
19:    for each individual  $Q_i \in Q$  do  $\triangleright$  We treat individuals as vectors below
20:       $\vec{a} \leftarrow$  a copy of an individual other than  $Q_i$ , chosen at random with replacement from  $Q$ 
21:       $\vec{b} \leftarrow$  a copy of an individual other than  $Q_i$  or  $\vec{a}$ , chosen at random with replacement from  $Q$ 
22:       $\vec{c} \leftarrow$  a copy of an individual other than  $Q_i$ ,  $\vec{a}$  or  $\vec{b}$ , chosen at random with replacement from  $Q$ 
23:       $\vec{d} \leftarrow \vec{a} + \alpha(\vec{b} - \vec{c})$   $\triangleright$  Mutation is just a arithmetic vector
24:       $P_i \leftarrow$  one child from  $Crossover(\vec{d}, Copy(Q_i))$ 
25:    end for
26:  end for
27: until  $Best$  is the ideal solution or we ran out of time
28: return  $b$ 

```

In the application of the ϵ DE algorithm in the water supply systems tested during the present work, violations arise from the non-observance of the equation of continuity of water level.

$$\text{if } L_{i,final} - L_{i,initial} \neq 0 \Rightarrow v1_i = L_{i,final} - L_{i,initial}, \forall i = 1, \dots, t. \quad (3.37)$$

The violation $v1$ is the difference between the initial level $L_{initial}$ and the final level L_{final} of each tank i .

Violations arise as well from disrespect of maximum tank levels:

$$\text{if } L_i - L_{i,max} > 0 \Rightarrow v2_i = L_i - L_{i,max}, \forall i = 1, \dots, t. \quad (3.38)$$

as well as from disrespect of minimum tank levels:

$$\text{if } L_i - L_{i,min} < 0 \Rightarrow v3_i = L_i - L_{i,min}, \forall i = 1, \dots, t. \quad (3.39)$$

The violations $v2$ and $v3$ are the difference between the actual water level, L_i , and the maximum level $L_{i,max}$ or minimum level, $L_{i,min}$, respectively, for each time-step i .

The total violation for each solution is the sum of the previous violations (equation 3.40).

$$v_i = v1_i + v2_i + v3_i, \forall i = 1, \dots, t. \quad (3.40)$$

For the implementation of this optimisation algorithm a C++ code, with around 900 lines of developed code. The developed code was based in a C code from the author of the algorithm [27]. The code was linked to the hydraulic simulation using the EPANET external libraries, allowing the calculation of both the objective function and the constraint violations needed to the optimisation by this algorithm.

3.4 Optimisation variables aggregation

In the presented methodology a new approach was followed in order to reduce the number of optimisation variables, simplifying the optimisation problem. This approach consisted in agglomeration of the optimisation variables taking into account the water demands and the energy tariff. During a certain period containing several time-steps, if it is attested that both water demand and energy tariff remain constant, then the correspondent time-steps can be aggregated into only one. This means that, for example, if four time-steps are available for aggregation, instead of eight optimisation variables (four time-steps with two optimisation variables per time-step and per pump) there will be considerate only two variables (one time-step with two optimisation variables per time-step and per pump).

3.5 HMI

To display the results obtained from the optimisation of the WSS it is proposed the development of an HMI. The development of the HMI, in this situation a GUI, followed 3 different steps: idealization, mock-up design and final design. The idea for this GUI was to reach a transparent and easy to understand interface. Learning time for new users should be almost instantaneous. Elements used in the interface should be common to

other software to ease user experience. To achieve this, the solution found intended to present results in a tabular scheme, each tab presenting different data. With the use of software *Balsamiq Mockups*, the initial mock-ups were developed. In figure 3.2 the initial screen of the GUI is presented. In this screen, the user select the type of optimisation and the network to optimize. The user gives the order to start the optimisation by clicking the button start.

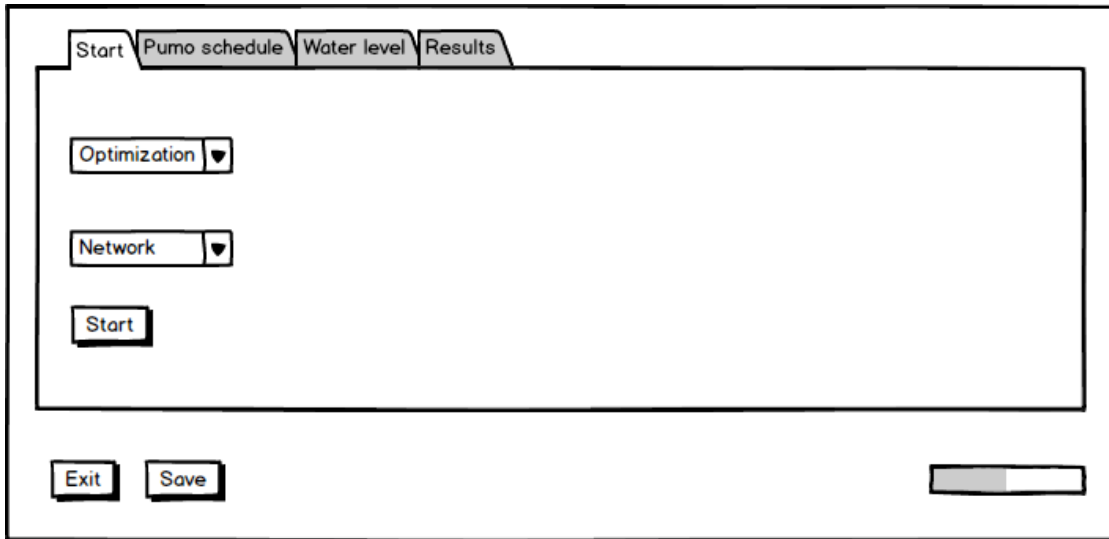


Figure 3.2: Mock-up of the initial screen of the GUI

In figure 3.3 the pump control screen of the GUI is presented. In this screen, the user is able to read information, graphically, about the control of the pumps.

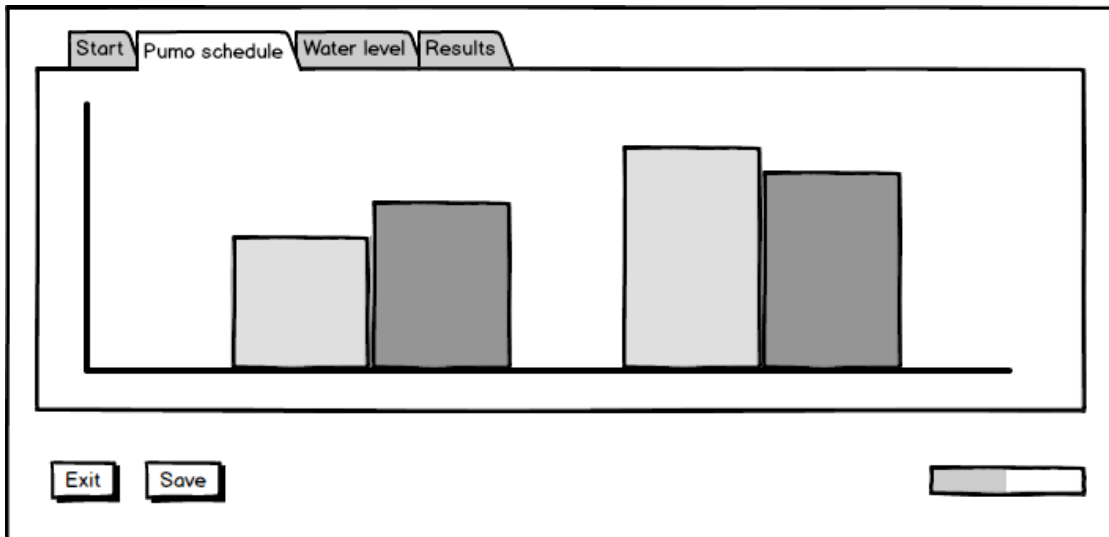


Figure 3.3: Mock-up of the pump control screen of the GUI

In figure 3.4 the pump control screen of the GUI is presented. In this screen, the user is able to read information, graphically, about the evolution of water level in the tanks, during the day.

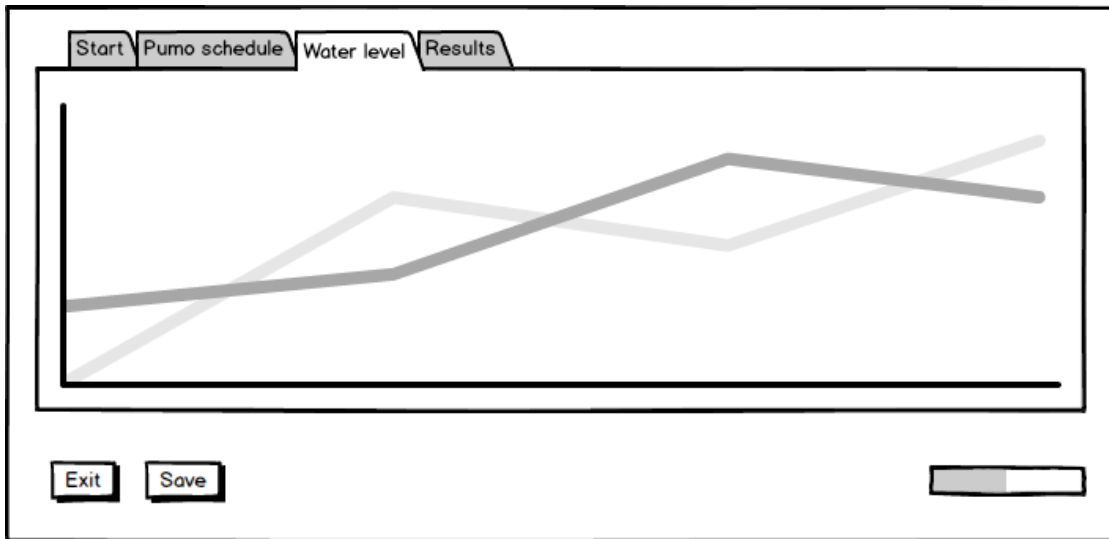


Figure 3.4: Mock-up of the water level screen of the GUI

In figure 3.4 the pump control screen of the GUI is presented. In this screen, the user is able to read information about the cost reductions reached by the algorithm used.

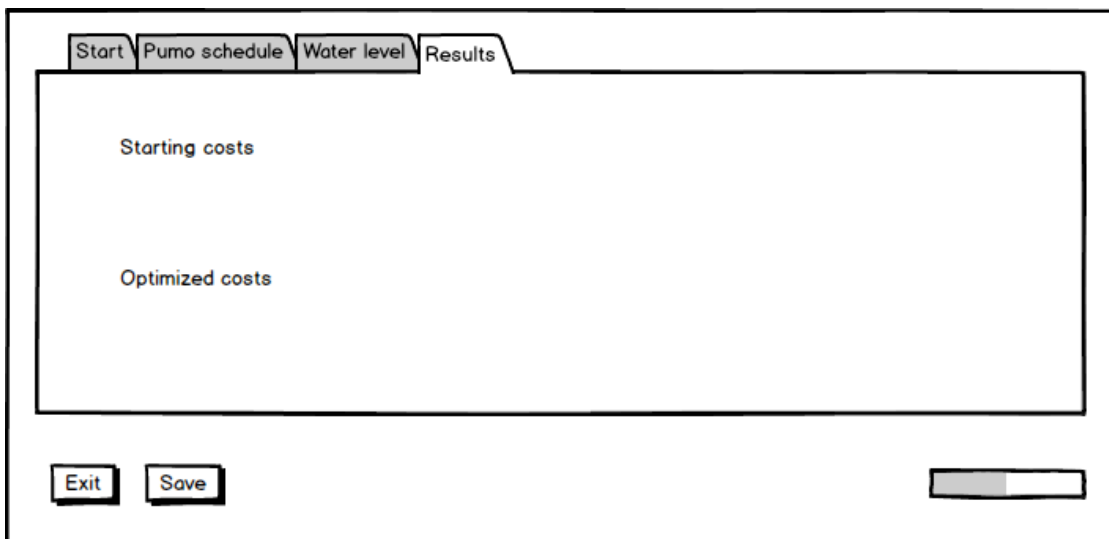


Figure 3.5: Mock-up of the final results screen of the GUI

3.6 Developed GUI

To develop the GUI, the software used was Visual Studio 2010. The selection of this software was based on the vast array of functionalities it possesses and its use would ease connection with the algorithms code, which was developed using the same software.

Based on the mock-ups previously made, presented in section 3.5, the GUI developed is displayed in this section. Figure 3.6 displays the starting page of the GUI (START

PAGE). The user selects the type of optimisation and the network to optimize with a combo box. After both are selected, the START button at the center of the GUI becomes active. After the user presses the start button, the optimisation takes place. This process can be stopped at any time pressing the button at the bottom right of the GUI. At the same location, exists a progress bar, allowing the user to access the progress of the optimisation.

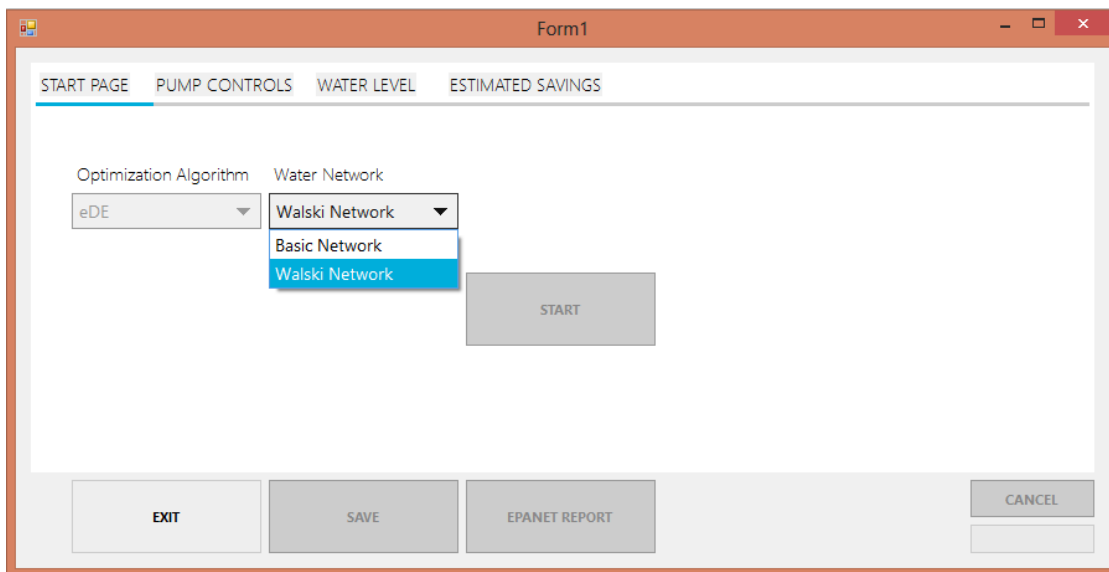


Figure 3.6: Interface Starting page

After the optimisation is finished, the tabs PUMP CONTROLS (figure 3.7), WATER LEVEL (figure 3.8) and ESTIMATED SAVINGS (figure 3.9). The button SAVE, which allows the user to save the results to a text file and the button EPANET REPORT, which opens the report file create by EPANET also become active.

Figure 3.7 shows the tab PUMP CONTROLS, where two bar plots display the usage of the pump, both with usage time and pump velocity. The existence of more pumps creates more tabs, one for each pump.

Figure 3.8 shows the tab WATER LEVEL, where the water level of a tank is displayed through the use of a chart . The existence of more tanks creates more tabs, one for each tanks.

Figure 3.9 shows the last tab, ESTIMATED SAVINGS. In this tab the user is presented with the cost value of the network prior to optimisation and after optimisation, to be able to evaluate the success of the operation.

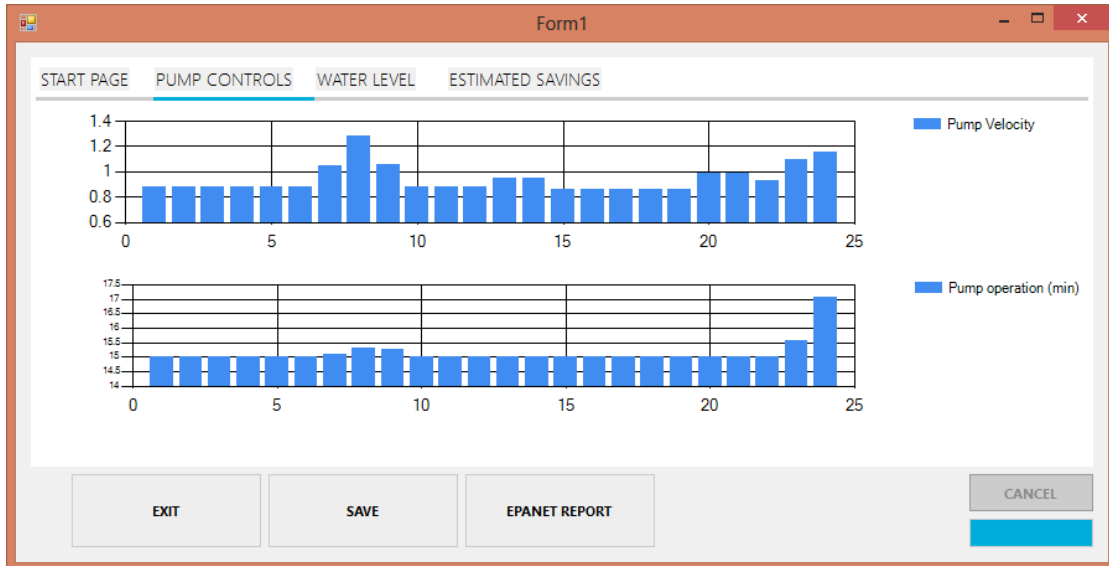


Figure 3.7: Interface Pump control page

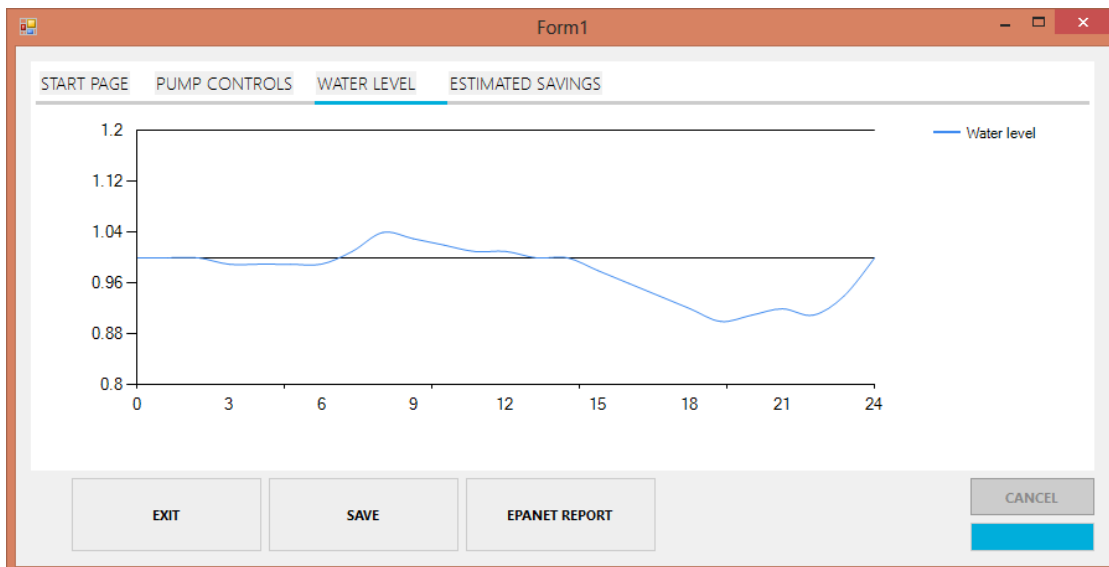
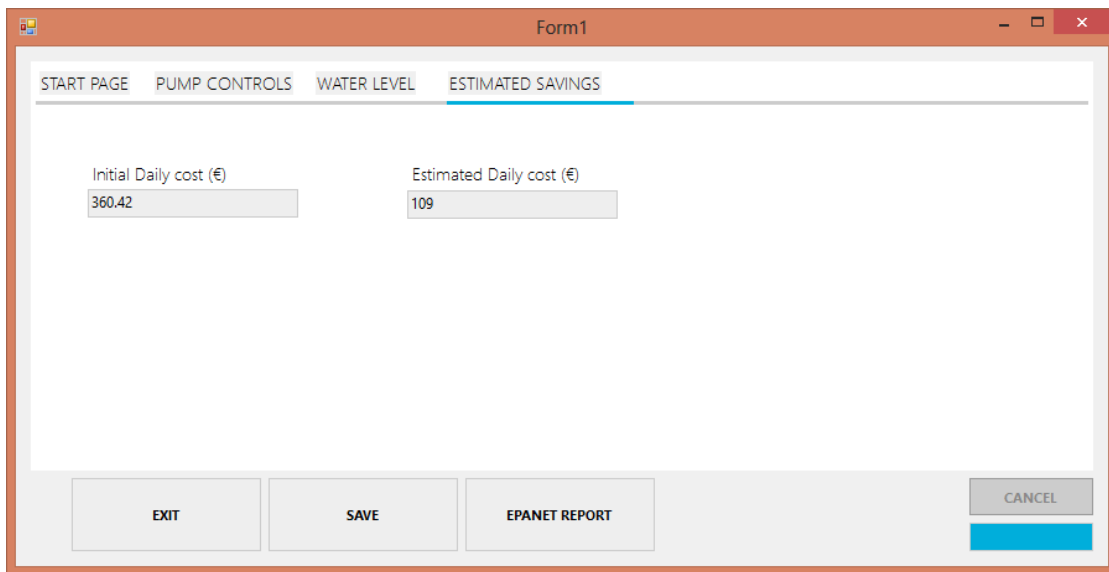


Figure 3.8: Interface Water level page



The screenshot shows a software window titled 'Form1' with a menu bar containing 'START PAGE', 'PUMP CONTROLS', 'WATER LEVEL', and 'ESTIMATED SAVINGS'. The 'ESTIMATED SAVINGS' page is active, showing two input fields: 'Initial Daily cost (€)' with the value 360.42 and 'Estimated Daily cost (€)' with the value 109. At the bottom of the window, there are four buttons: 'EXIT', 'SAVE', 'EPANET REPORT', and 'CANCEL'. The 'CANCEL' button is highlighted in blue.

Figure 3.9: Interface Estimated savings page

The user can start another optimisation, by simply selecting another option at START PAGE and pressing the START button.

Part III
Results

Chapter 4

Numerical Results

4.1 Benchmarks

To validate future results obtained by the implemented algorithms, those must be previously validated using standard test functions. This method allows testing the performance of mentioned algorithms. Although these functions are unconstrained, unlike the WSS that are object of study in this work, these functions can give input of the correct functioning of the algorithms.

4.1.1 Test Functions

De Jong's function

The *De Jong's* first equation is also known as the sphere equation. It's one of the simplest test benchmarks and was first proposed by Kenneth Alan de Jong [28]. The function is continuous, convex and unimodal. The general definition of the function is given by:

$$f(x) = \sum_{i=1}^n x_i^2. \quad (4.1)$$

The solution space is usually restricted to a hypercube defined as $-5.12 \leq x_i \leq 5.12$, for $i = 1, \dots, n$. The global minimum of the function $f(x) = 0$ is obtained at $x_i = 0, i = 1, \dots, n$. A 3D representation of the *De Jong function* with $n = 2$ is shown in figure 4.1.

Axis parallel hyper-ellipsoid function

Also known as the *weighted sphere model*, this function is similar to *De Jong function*. The function is continuous, convex and unimodal. The general definition of the function is presented in:

$$f(x) = \sum_{i=1}^n ix_i^2. \quad (4.2)$$

The search universe is usually also restricted to a hypercube defined as $-5.12 \leq x_i \leq 5.12$, for $i = 1, \dots, n$. The global minimum of the function $f(x) = 0$ is obtained at $x_i = 0, i = 1, \dots, n$. A Three Dimensions (3D) representation of the *Axis parallel hyper-ellipsoid function* with $n = 2$ is shown in figure 4.2.

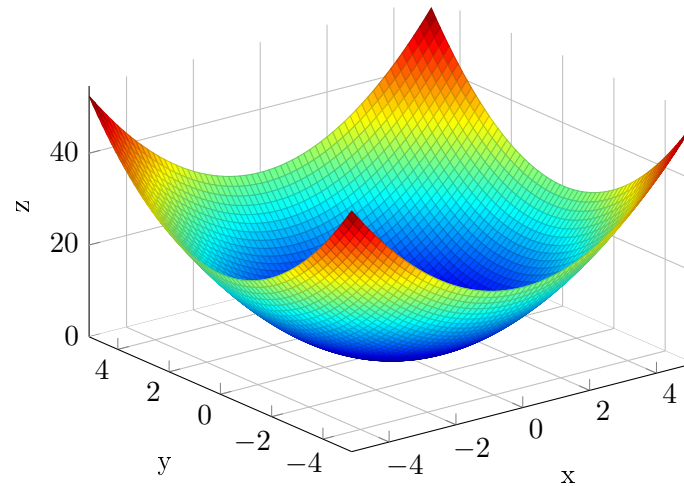


Figure 4.1: 3D representation of the De Jong function for the hypercube defined as $-5.12 \leq x_i \leq 5.12$ and $i = 2$.

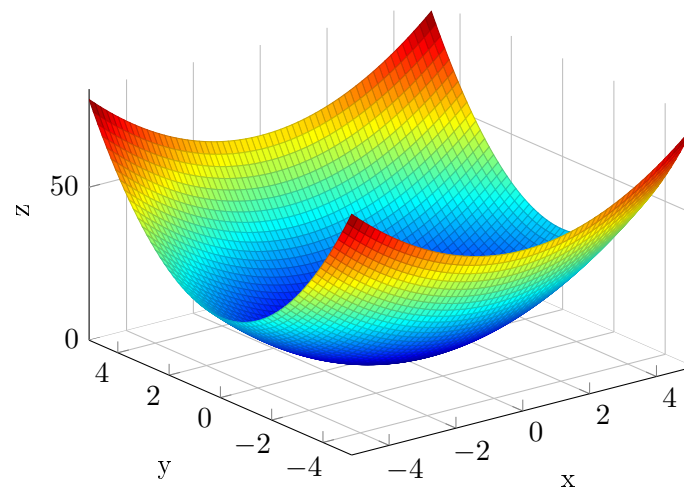


Figure 4.2: 3D representation of the Axis parallel hyper-ellipsoid function for the hypercube defined as $-5.12 \leq x_i \leq 5.12$ and $i = 2$.

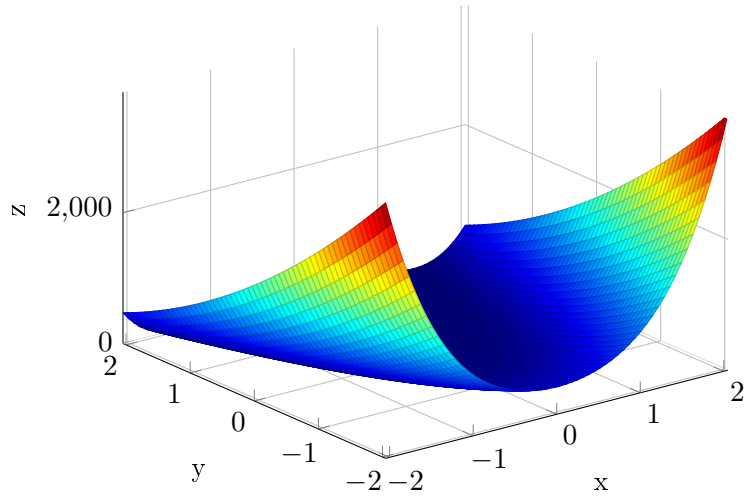


Figure 4.3: 3D representation of the Rosenbrock valley problem for the hypercube defined as $-2.048 \leq x_i \leq 2.048$ and $i = 2$.

Rosenbrock's valley

Also known as the *banana function* or the *second function of De Jong*, the *Rosenbrock valley problem* is a classic optimisation problem that was first proposed by Kenneth Alan de Jong [28]. The general definition of the function is presented in equation 4.3:

$$f(x) = \sum_{i=1}^{n-1} \left[100 \times (x_{i-1} - x_i^2)^2 + (1 - x_i)^2 \right] \quad (4.3)$$

The search area is usually restricted to a hypercube defined as $-2.048 \leq x_i \leq 2.048$, for $i = 1, \dots, n$. The global minimum of the function $f(x) = 0$ is obtained at $x_i = 0, i = 1, \dots, n$. The minimum lies inside a long, narrow and parabolic shaped flat valley, which can be seen in figure 4.3. This problem causes difficulties because, while finding the valley is trivial, to find the global minimum is difficult. A 3D representation of the *Rosenbrock's valley problem* with $n = 2$ is shown in figure 4.3.

Easom's Function

The Easom function is an optimisation problem where the global minimum has a small area, relatively to the search space. The function is unimodal, and its general mathematical definition is presented as:

$$f(x_1, x_2) = -\cos(x_1) \cos(x_2) \exp \left[-(x_1 - \pi)^2 - (x_2 - \pi)^2 \right]. \quad (4.4)$$

The test area is usually restricted to a hypercube defined as $-100 \leq x_1 \leq 100, -100 \leq x_2 \leq 100$. The global minimum of the function $f(x) = 0$ is obtained at $(x_1, x_2) = (\pi, \pi)$. A 3D representation of the *Easom's Function* is shown in figure 4.4.

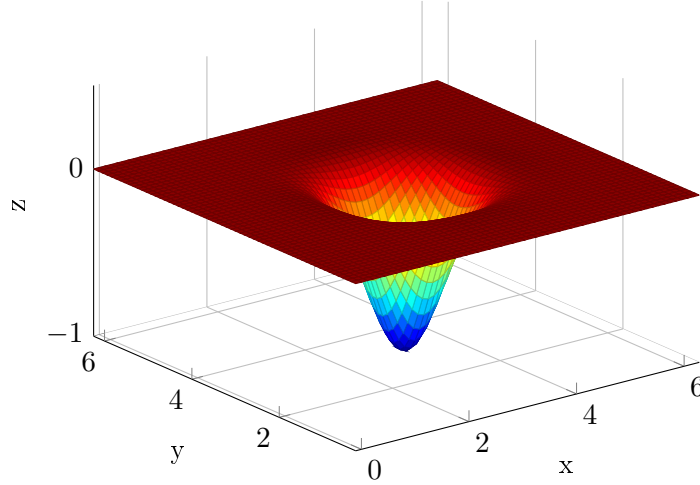


Figure 4.4: 3D representation of Easom's function, represented in the space $0 \leq x \leq 2\pi$ and $0 \leq y \leq 2\pi$.

Rastrigin Function

Rastrigin's function is a variation of the original *De Jong* function, to which is added cosine modulation in order to produce frequent local minima and maxima. The Rastrigin Function is highly multimodal, but the distribution of the local extremes is regularly distributed. The general definition of the function is presented in equation 4.5:

$$f(x) = \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i) + 10). \quad (4.5)$$

Equally to *De Jong Function*, the search space is usually restricted to a hypercube defined as $-5.12 \leq x_i \leq 5.12$, and $i = 1, \dots, n$. The global minimum of the function $f(x) = 0$ is obtained at $x_i = 0, i = 1, \dots, n$. A 3D representation of the *Rastrigin Function* is shown in figure 4.5.

Ackley's Function

Ackley's Function is a widely used optimisation test problem. It is a multimodal problem. The general definition of the function is written as:

$$f(x) = -20 \exp \left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2} \right) - \exp \left(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i) \right) + 20 + e \quad (4.6)$$

The test area is usually restricted to a hypercube defined as $-32.768 \leq x_i \leq 32.768$, and $i = 1, \dots, n$. The global minimum of the function $f(x) = 0$ is obtained at $x_i = 0, i = 1, \dots, n$. A 3D representation of the *Ackley's Function* is shown in figure 4.6.

Schwefel's Function

Schwefel's Function is an optimisation problem that can cause problems to tested algorithms for its deceptive behaviour, as the global minima is geometrically distant to the

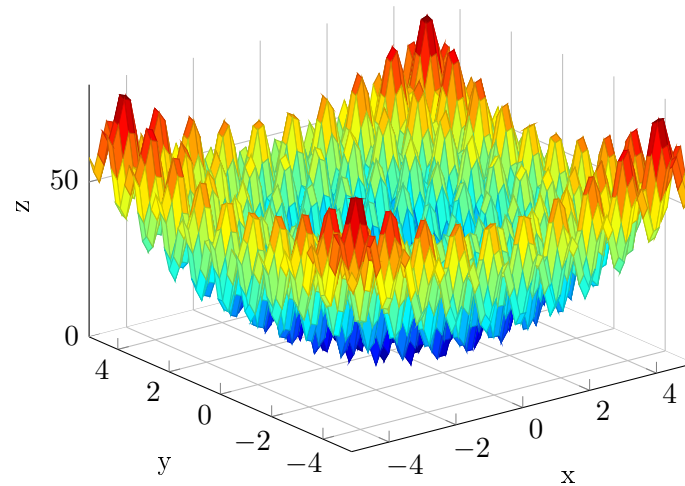


Figure 4.5: 3D representation of the Rastrigin Function for the hypercube defined as $-5.12 \leq x_i \leq 5.12$ and $i = 2$.

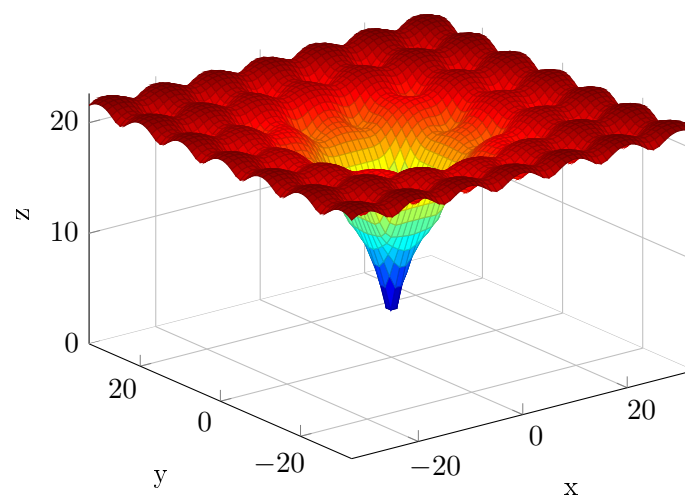


Figure 4.6: 3D representation of the Ackley's Function for the hypercube defined as $-32.768 \leq x_i \leq 32.768$ and $i = 2$.

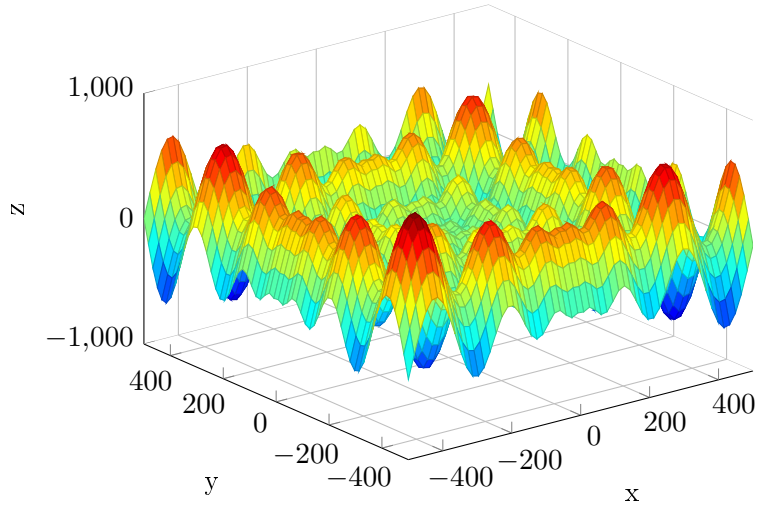


Figure 4.7: 3D representation of the Schwefel's Function for the hypercube defined as $-500 \leq x_i \leq 500$ and $i = 2$.

next best local minima. This can cause problems of convergence to some algorithms. The general definition of the function is given by:

$$f(x) = \sum_{i=1}^n \left[-x_i \sin(\sqrt{|x_i|}) \right]. \quad (4.7)$$

The test area is usually restricted to a hypercube defined as $-500 \leq x_i \leq 500$, and $i = 1, \dots, n$. The global minimum of the function $f(x) = -418.9829n$ is obtained at $x_i = 420.9687, i = 1, \dots, n$. A 3D representation of the *Schwefel's Function* is shown in figure 4.7.

Michalewicz's Function

The Michalewicz's Function is a multimodal optimisation problem. The general definition of the function is presented as:

$$f(x) = - \sum_{i=1}^n \sin(x_i) \left[\sin\left(\frac{ix_i^2}{\pi}\right) \right]^{2m}. \quad (4.8)$$

The test area is usually restricted to a hypercube defined as $0 \leq x_i \leq \pi$, and $i = 1, \dots, n$. It is usually set $m = 10$. The global minimum of the function as been approximated by $f(x) = -4.687$ for $n = 5$ and $f(x) = -9.66$ for $n = 10$. A 3D representation of the *Michalewicz's Function* is shown in figure 4.8.

4.1.2 Benchmark Results

The representation of the optimisation formulation for each of the following functions can be given by:

$$\begin{aligned} & \text{minimize} && f(\mathbf{x}), \\ & \text{subject to} && \mathbf{x}_i^{\min} < \mathbf{x}_i < \mathbf{x}_i^{\max}, \end{aligned} \quad (4.9)$$

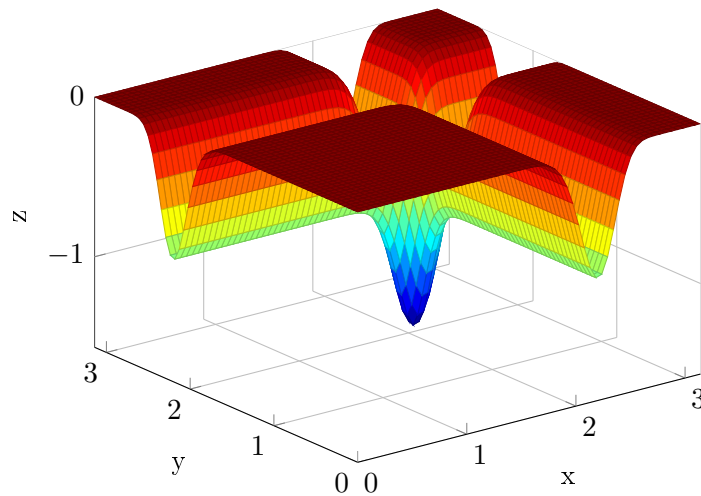


Figure 4.8: 3D representation of the Michalewicz's Function for the hypercube defined as $0 \leq x_i \leq \pi$ and $i = 2$.

where $f(x)$ is the objective function and $i = 1, \dots, t$ is the number of optimisation variables.

For each function, to solve the proposed optimisation problem the ϵ DE method was used with a population of 50 individuals, a scaling factor of 0.7, a crossover rate of 0.9 and a maximum number of generations of 50 generations

Equally, using the L-BFGS-B method to solve the optimisation problem, the given disturbance value for the gradient was 0.0001. The value given to the factor associated with the stopping criteria was $1.0e^7$.

De Jong's function

To the optimisation of the De Jong's function were used 20 optimisation variables, with $-5.12 \leq x_i \leq 5.12$. The results for the De Jong's function optimisation are displayed in table 4.1 and in figure 4.9. The obtained results show a quicker convergence by the L-BFGS-B method, and good results on the final value by both methods. These results were expected, as the L-BFGS-B method thrives in convex nature functions. The value of the De Jong function at the start of the ϵ DE optimisation is different from the value of De Jong's function at the start of the L-BFGS-B because the starting point for the ϵ DE is random, while for the L-BFGS-B the starting value was $x_i = 0.5$.

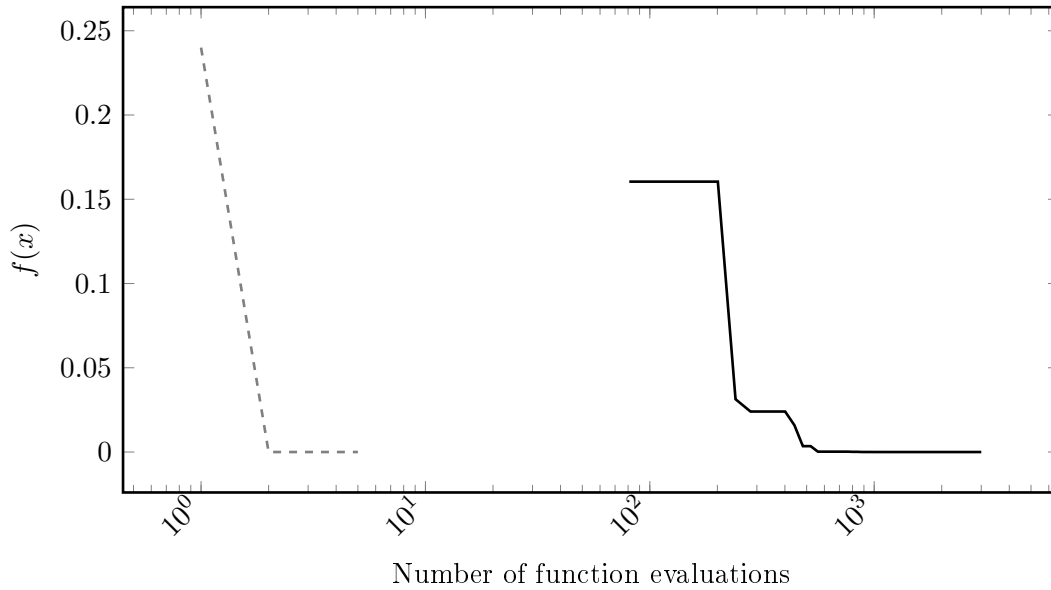


Figure 4.9: Evolution of $f(x)$ of De Jong's function when using the ε DE method (in black) and the L-BFGS-B method (in dashed gray).

Table 4.1: optimisation results of De Jong's function for the ε DE method and the L-BFGS-B method.

	Optimum value	Evaluations
Theoretical	0	
ε DE	4.83E-17	3000
L-BFGS-B	3.75E-23	5

Axis parallel hyper-ellipsoid function

The optimisation of the Axis parallel hyper-ellipsoid function was made with 20 optimisation variables, with $-5.12 \leq x_i \leq 5.12$. The results for the Axis parallel hyper-ellipsoid function optimisation are displayed in table 4.2 and in figure 4.10. The obtained results show a quicker convergence by the L-BFGS-B method and good results on the final value by both methods. These were the expected results, as the convex nature of the function benefits the gradient search method applied by the L-BFGS-B. The starting values of both the ε DE optimisation and the L-BFGS-B optimisation are different because while the starting point for the ε DE is random, for the L-BFGS-B the starting value was $x_i = 0.5$.

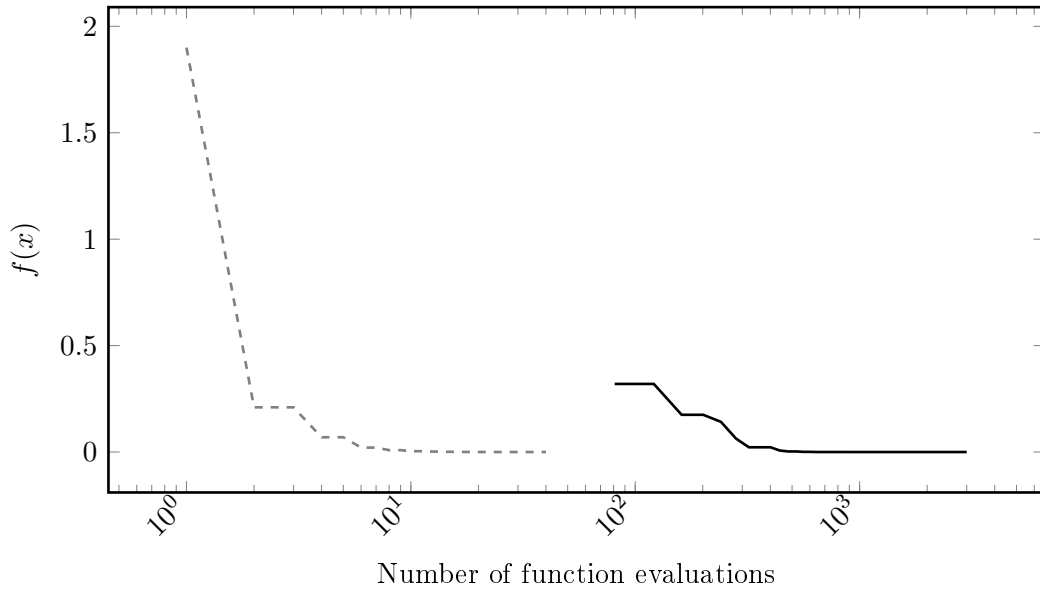


Figure 4.10: Evolution of $f(x)$ of the Axis parallel hyper-ellipsoid function when using the ε DE method (in black) and the L-BFGS-B method (in dashed gray).

Table 4.2: optimisation results of Axis parallel hyper-ellipsoid function for the ε DE method and the L-BFGS-B method.

	Optimum value	Evaluations
Theoretical	0	
ε DE	4.53E-17	3000
L-BFGS-B	6.32E-10	40

Rosenbrock's valley

The optimisation of the Rosenbrock's valley function was made with 20 optimisation variables, with $-2.048 \leq x_i \leq 2.048$. The results for the Rosenbrock's valley function optimisation are displayed in table 4.3 and in figure 4.11. The obtained results show a quicker convergence by the L-BFGS-B method, and good results on the final value by both methods. Figure 4.11 shows that the L-BFGS-B method converge very fast to the valley of the function, while the refined search for the optimum point takes more evaluations. This is due to the low derivative of the function within the valley. As with the previous functions, the difference of the starting points for the ε DE method and the L-BFGS-B is due to the random starting points in the ε DE, while for the L-BFGS-B the starting point was $x_i = 0.5$.

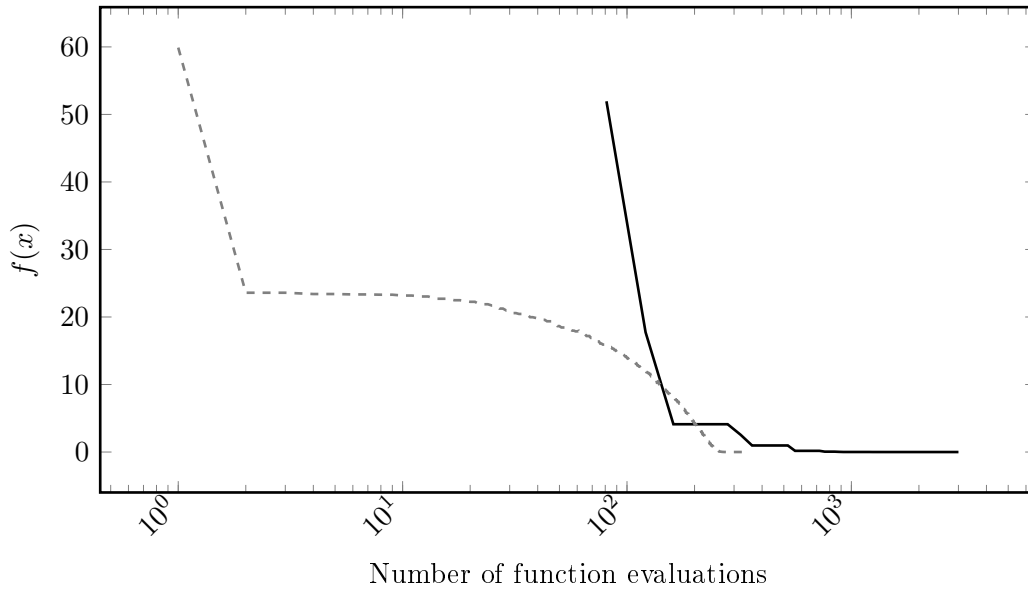


Figure 4.11: Evolution of $f(x)$ of the Rosenbrock's valley function when using the ε DE method (in black) and the L-BFGS-B method (in dashed gray).

Table 4.3: optimisation results of Rosenbrock's function for the ε DE method and the L-BFGS-B method.

	Optimum value	Evaluations
Theoretical	0	
ε DE	4.68E-13	3000
L-BFGS-B	2.30E-03	328

Easom's Function

The optimisation of the Easom's Function was made with 2 optimisation variables, with $-100 \leq x_1 \leq 100, -100 \leq x_2 \leq 100$.

The results for the Easom's function optimisation are displayed in table 4.4 and in figure 4.12. The obtained results show convergence by the ε DE method, but the L-BFGS-B method is unable to reach any results. The nature of the function, whose 3D representation is presented in subsection 4.1.1 (figure 4.4), limits the feasibility of the L-BFGS-B method to the small area around the optimum point. Outside that area, the derivative of the function is 0, not giving any useful information to the progress of the L-BFGS-B method.

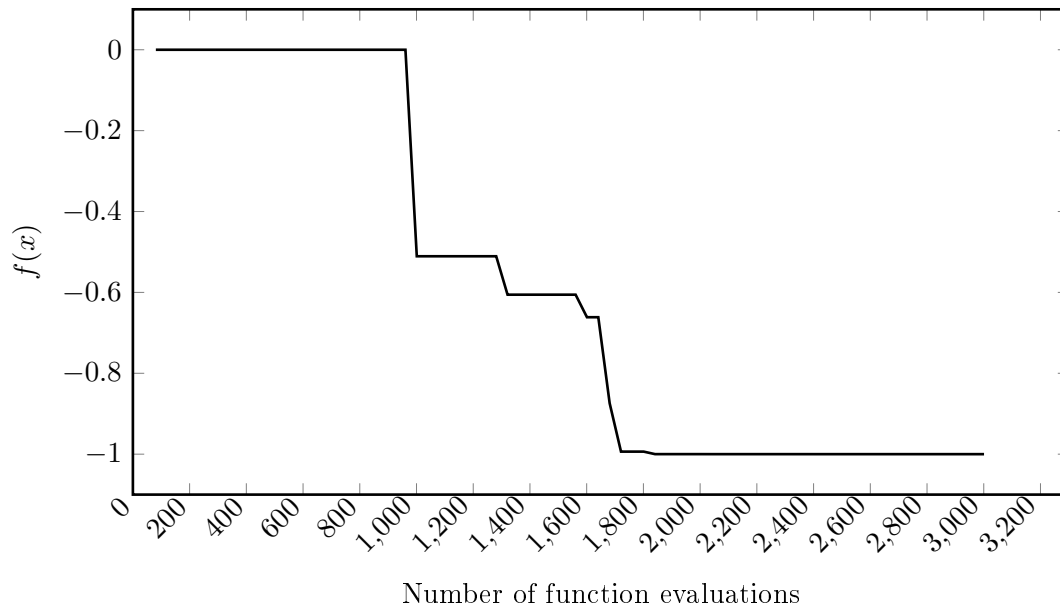


Figure 4.12: Evolution of $f(x)$ of the Easom's function when using the ε DE method (in black).

Table 4.4: optimisation results of Easom's function for the ε DE method and the L-BFGS-B method.

	Optimum value	Evaluations
Theoretical	-1	
ε DE	-0.99	3000
L-BFGS-B	n/a	n/a

Rastrigin Function

For the optimisation of the Rastrigin function were used 20 optimisation variables, with $-5.12 \leq x_i \leq 5.12$. The results for the Rastrigin function optimisation are displayed in table 4.5 and in figure 4.13. The obtained results show convergence by the ε DE method, however the L-BFGS-B method is unable to reach results. The nature of the function, whose 3D representation is presented in subsection 4.1.1 (figure 4.5), limits the feasibility of the L-BFGS-B method as it has many local minima. As it only works with a starting point near the optimum value, no useful information about the progress of the L-BFGS-B method is obtained.

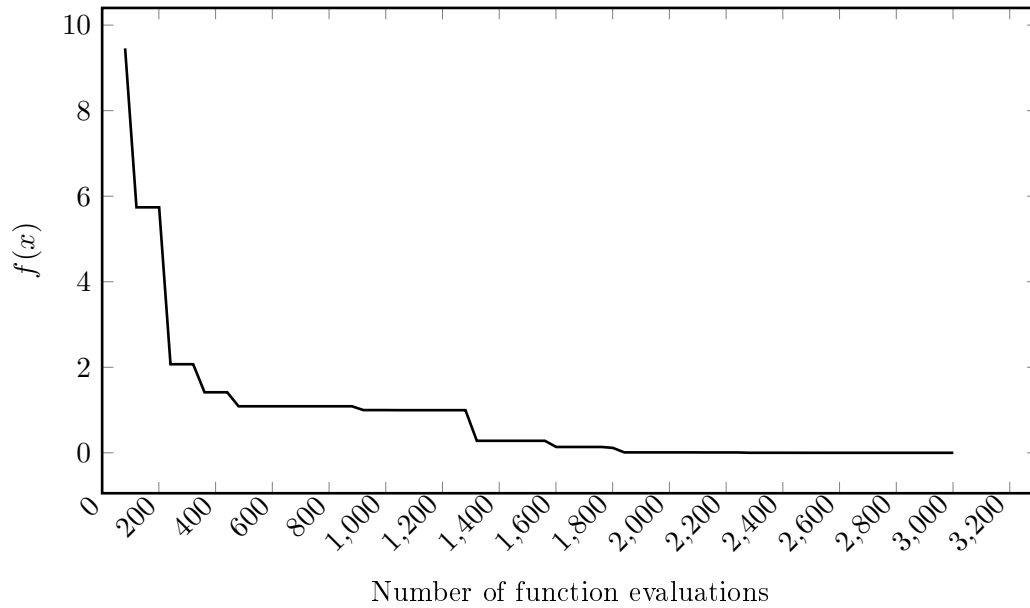


Figure 4.13: Evolution of $f(x)$ of the Rastrigin function when using the ε DE method (in black).

Table 4.5: optimisation results of Rastrigin function for the ε DE method and the L-BFGS-B method.

	Optimum value	Evaluations
Theoretical	0	
ε DE	2.83E-08	3000
L-BFGS-B	n/a	n/a

Ackley's Function

The optimisation of the Axis parallel hyper-ellipsoid function was made with 20 optimisation variables, with $-32.768 \leq x_i \leq 32.768$. The results for the Ackley function optimisation are displayed in table 4.6 and in figure 4.14. The obtained results show convergence by the ε DE method, but the L-BFGS-B method is unable to reach results. Ackley's function, whose 3D representation is presented in subsection 4.1.1 (figure 4.6), has many local minima. The L-BFGS-B method applied to this function gives no useful information about the progress of the algorithm.

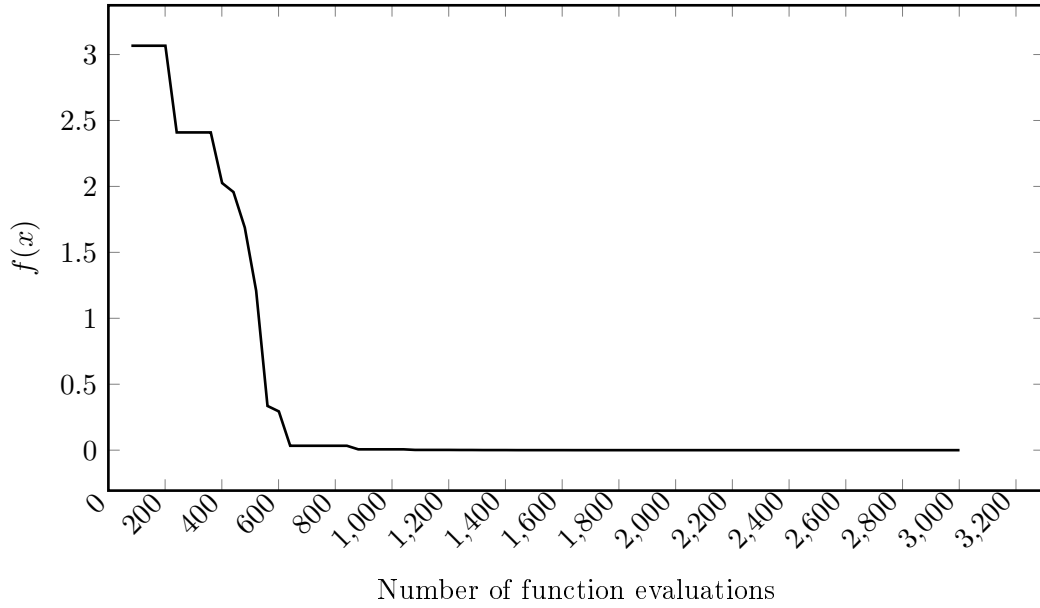


Figure 4.14: Evolution of $f(x)$ of the Ackley's function when using the ε DE method (in black).

Table 4.6: optimisation results of Ackley's function for the ε DE method and the L-BFGS-B method.

	Optimum value	Evaluations
Theoretical	0	
ε DE	2.23E-08	3000
L-BFGS-B	n/a	n/a

Schwefel's Function

To the optimisation of the De Jong's function were used 10 optimisation variables, with $-500 \leq x_i \leq 500$.

The results for the Schwefel's function optimisation are displayed in table 4.7 and in figure 4.15. The obtained results show convergence by the ε DE method. However, the L-BFGS-B method is unable to reach results. The nature of the function, which 3D representation is presented in subsection 4.1.1 (figure 4.7), limits the feasibility of the L-BFGS-B method as it has a deceptive behaviour, as the global minima of the function is geometrically distant to the next best local minima. The function also has many local minima. For this the use of the L-BFGS-B method does not give information about the progress of the algorithm.

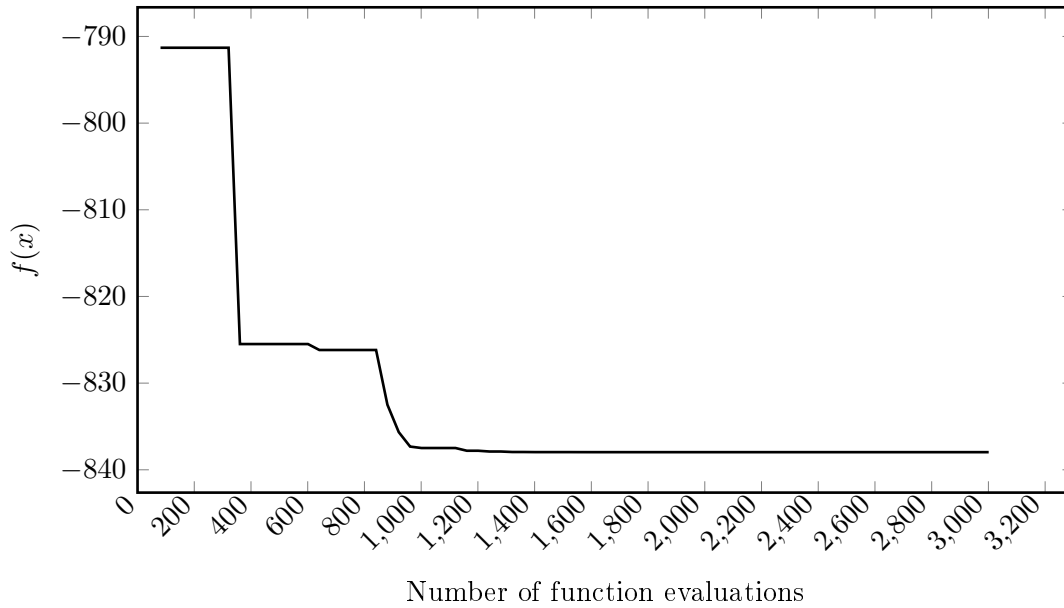


Figure 4.15: Evolution of $f(x)$ of the Schwefel's function when using the ε DE method (in black).

Table 4.7: optimisation results of Schwefel's function for the ε DE method and the L-BFGS-B method.

	Optimum value	Evaluations
Theoretical	-837.9658	
ε DE	-837.9658	3000
L-BFGS-B	n/a	n/a

Michalewicz's Function

For the optimisation of the Michalewicz's function were used 5 optimisation variables, with $0 \leq x_i \leq \pi$. The value m was set at 10. The results for the Michalewicz's function optimisation are displayed in table 4.8 and in figure 4.16. The obtained results show convergence by the ε DE method. The L-BFGS-B method, however, is unable to reach results. The nature of the function, which 3D representation is presented in subsection 4.1.1 of this work, figure 4.8, limits the feasibility of the L-BFGS-B method as it has null derivatives in the most part of the solution, not giving any useful information to the progress of the L-BFGS-B method.

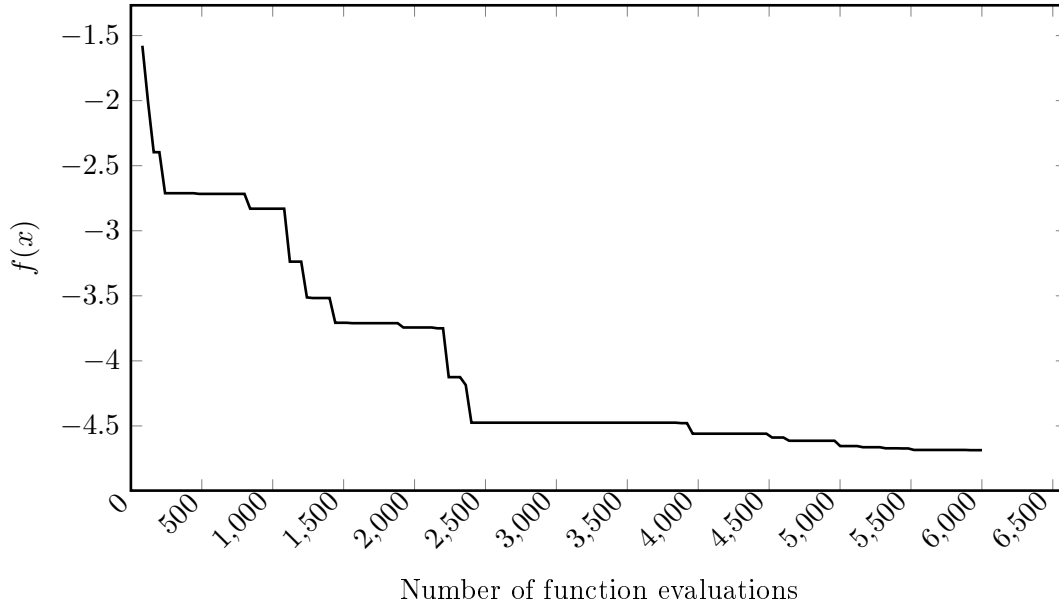


Figure 4.16: Evolution of $f(x)$ of the Michalewicz's function when applying the constrained differential algorithm

Table 4.8: optimisation results of Michalewicz's function for the ε DE method and the L-BFGS-B method.

	Optimum value	Evaluations
Theoretical	-4.687	
ε DE	-4.686	6000
L-BFGS-B	n/a	n/a

4.2 Basic Network

4.2.1 Network Modelling

A simple network, represented in figure 4.17, selected from the study of Coelho *et al.* [29] was used to evaluate the algorithms proposed in this work. This network is composed of a reservoir, R, with an elevation of 50 meters, a pump, P, responsible for the pumping of water from the reservoir to a variable level tank, T, with 400 meters of elevation. Tank T has a diameter of 40 meters, a minimum level of 0.5 meters and a maximum level of 5 meters. The initial level in tank T is 1 meter. A consumption node, D, has an elevation of 300 meters and is supplied by the variable level tank. The base demand of the consumption node D is 10 l/s, with the demand pattern for a day represented in figure 4.18. The energy costs considered in this problem are represented in figure 4.19 in which the tariffs for each hour are shown. The characteristic curve of the pump is presented in figure 4.20, with the relative speeds considered in this work ($\omega = 0.5$ and $\omega = 2$) included. Initial values of the network modelling are presented in table 4.9. The

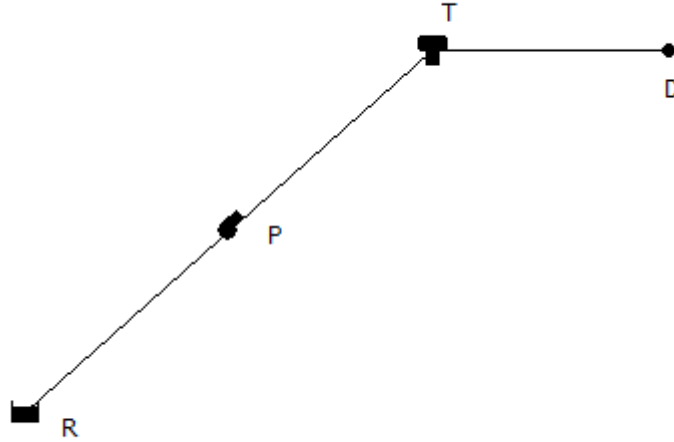


Figure 4.17: Scheme of Basic network.

total cost, for a simulation with a duration of 24 periods of 1 hour each, was shown to be of 360,42€.

Table 4.9: Initial values of energy and cost for the Basic Network

Energy Consumption (kWh/m ³)	Average Power (kW)	Maximum Power (kW)	Daily Cost (€)
1.27	134.79	136.48	360.42

4.2.2 Results Comparison

The optimisation problem for the Basic network can be represented by:

$$\begin{aligned}
 & \text{minimize} && f(\mathbf{x}) = \text{energy cost}, \\
 & \text{subject to} && h(\mathbf{x}) = \text{continuity condition} = 0, \\
 & && g(\mathbf{x}) = \text{levels limits} \leq 0, \\
 & && \mathbf{x}_i^{\min} < \mathbf{x}_i < \mathbf{x}_i^{\max},
 \end{aligned} \tag{4.10}$$

where $f(x)$ is the objective function, calculated with the use of EPANET, $i = 1, \dots, 48$ is the number of optimisation variables, $h(x)$ are equality constraints and $g(x)$ are inequality constraints. For the variables of time the pump is on at each time step, the values between 0 and 1 obtained from the optimisation can be transformed to minutes following a linear equation:

$$\text{time} = x_i \times 60.$$

For the variables of pump speed, the values obtained from the optimisation can be transformed to the relative speed of the pump ω following a linear equation:

$$\omega = 0.5 + (x_i \times 1.5).$$

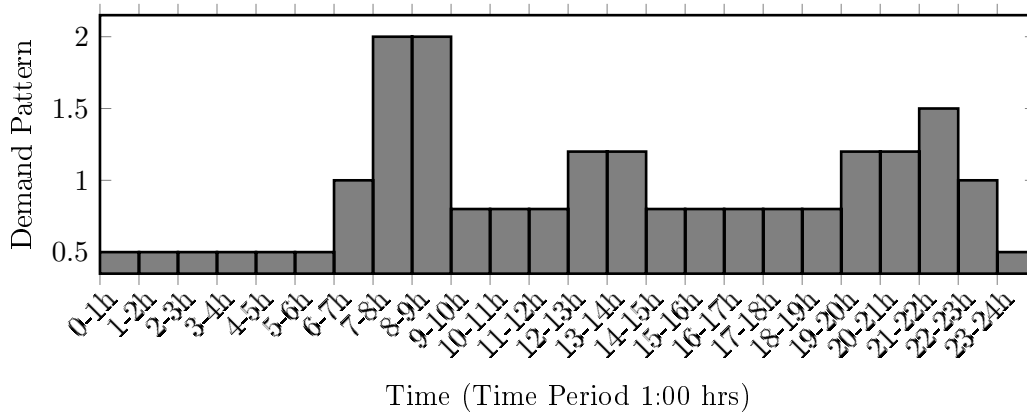


Figure 4.18: Consumption pattern associated with Basic Network.

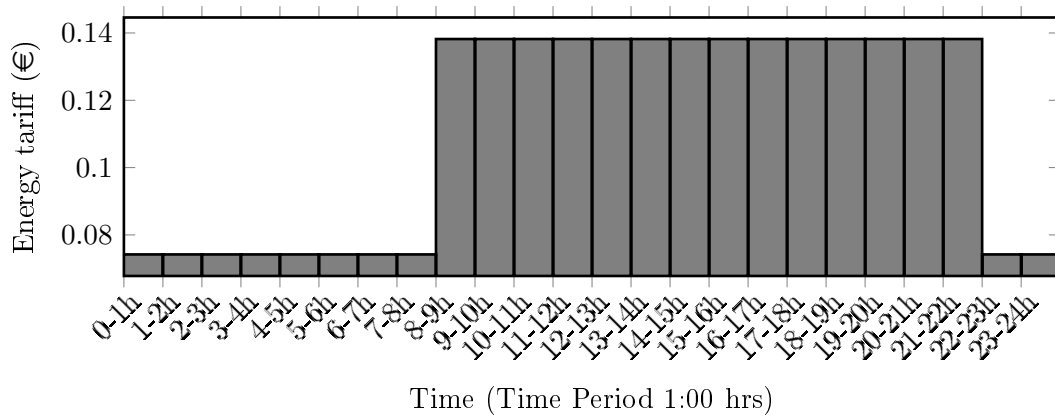
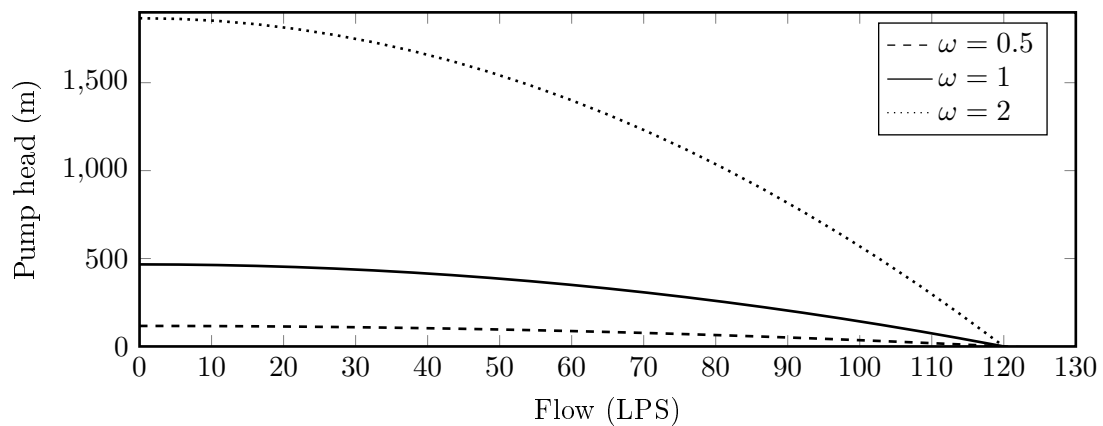


Figure 4.19: Energy tariff (€) associated with Basic Network.

Figure 4.20: Characteristic curve of the pump used in the Basic Network problem, with the representation for three distinct relative speeds: the nominal speed ($\omega = 1$), the minimum speed ($\omega = 0.5$) and the maximum speed ($\omega = 2$)

The optimisation problem is subjected to the following equality constraint:

$$h(x) = L_{final} - L_{initial} = 0, \quad (4.11)$$

with $L_{initial}$ being the initial water level and L_{final} the final water level of the tank.

The optimisation problem is subjected to the following inequality constraint:

$$g1(x) = L - L_{max}, \quad (4.12)$$

$$g2(x) = L - L_{min}, \quad (4.13)$$

with L being the current water level, $L_{max} = 6$ being the maximum admitted level and $L_{min} = 0.5$ the minimum admitted level for the tank.

To solve the proposed optimisation problem the ε DE method was used with a population of 50 individuals, a scaling factor of 0.7, a crossover rate of 0.9 and a maximum number of generations of 500 generations.

Using the L-BFGS-B method to solve the optimisation problem, the given disturbance value for the gradient was 0.0001. The value given to the factor associated with the stopping criteria was $1.0e^7$. After aggregation, the number of optimisation variables is 22.

In table 4.10 the obtained results of the optimisation of the Basic network presented in subsection 4.2.1 by the ε DE method, the DE method and the optimisation by the L-BFGS-B method, with a comparison of number of evaluation, CPU time, cost and cost reduction between the mentioned methods are presented. The DE method used as comparison uses the exterior penalties method to include restrictions in its evaluations.

Table 4.10: optimisation results of Basic network benchmark for the DE method, the ε DE method and the L-BFGS-B method.

Method	Evaluations	CPU time (min)	Cost €/day	Cost reduction (%)
ε DE	25000	19.8	82	77.2
DE	30000	28	70.9	80.3
L-BFGS-B	52	0.6	73.01	79.7

While all methods achieved significant cost reductions (over 75 %) the best results were obtained by the DE method. The results obtained from ε DE method are limited by the constrains previously presented, since the obtained solution does not incur in violations, and for that only solutions that have no error are considered by the method. The L-BFGS-B method requires only a small number of evaluations to achieve the final value. The evolution of the objective function for the L-BFGS-B method is compared to the evolution of the objective function for the ε DE method in figure 4.21. From the observance of figure 4.21 is visible an increment of the objective function. This is due to the nature of the ε DE method, which takes privilege to a decrease in the total error, independently of the growth of the objective function.

The final solution obtained from the ε DE respects all constraints. Such situation can be confirmed from figure 4.22. Both minimum and maximum levels of the tank are respected, and the continuity equations is verified as well. The flows at the pump necessary for this solution are also present in figure 4.22. In figure 4.23 it is observable the controls given to the pumps after the optimisation. Comparing the pump activity

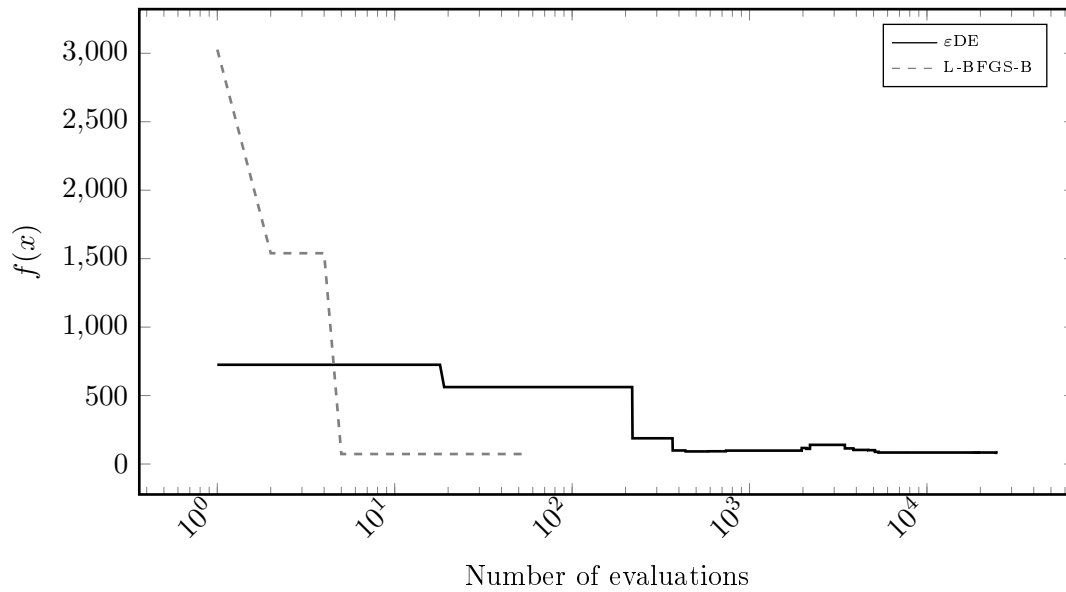


Figure 4.21: Cost function evolution throughout the optimisation process for Basic Network using the ϵ DE method (in black) and the L-BFGS-B method (in dashed gray) .

(figure 4.22 and 4.23) with the energy tariff (figure 4.19) it is observable that the pumps are active in the time periods just before the increase of energy cost (from 6h to 8h) and start again just after the energy tariff lower (from 22 h to 24h).

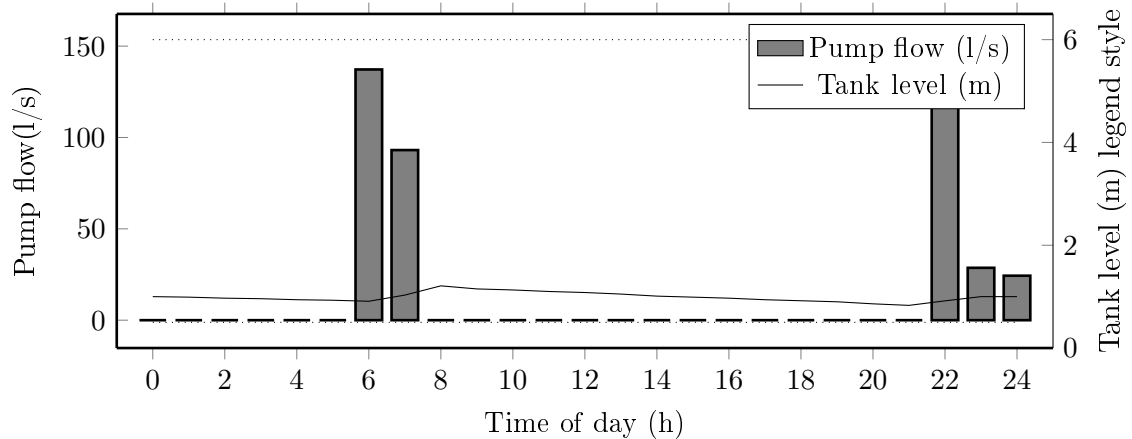


Figure 4.22: Pump flow (l/s) and tank level variation (m) during a day, after optimisation by the ϵ DE method.

The final solution obtained from the L-BFGS-B respects both the maximum and minimum level constraints. The continuity constraint is not achieved, although by only a small difference. The flows at the pump necessary for this solution are also present in figure 4.24. In figure 4.25 it is observable the controls given to the pumps after the optimisation. In this optimisation process, the pump is active only in the first hour of the day. The cost reduction is obtained with changes in the schedule of operation of the pump.

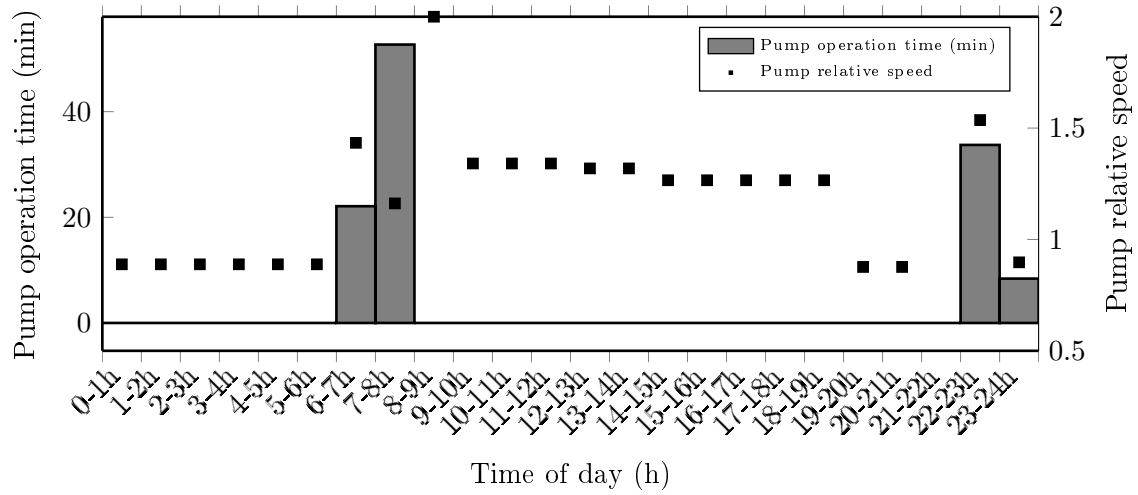


Figure 4.23: Pump controls (pump speed and operation time) during the day, after optimisation by the ε DE method.

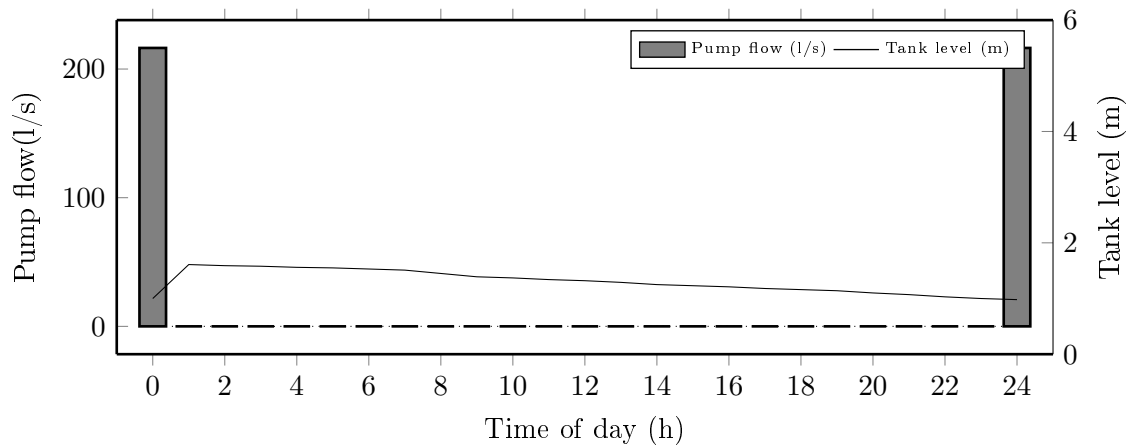


Figure 4.24: Pump flow (l/s) and tank level variation (m) during a day, after optimisation by the L-BFGS-B method.

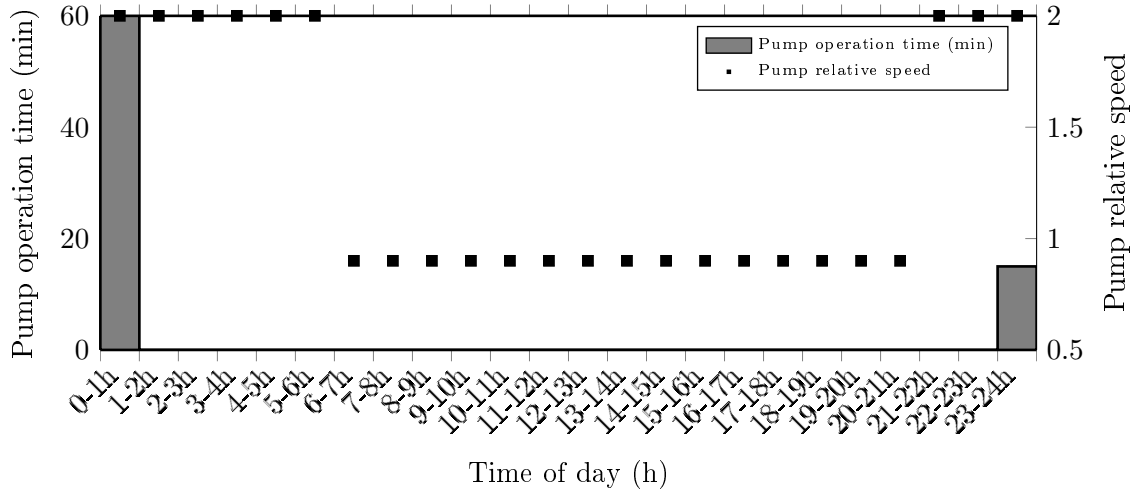


Figure 4.25: Pump controls (pump speed and operation time) during the day, after optimisation by the L-BFGS-B method.

4.3 Walski Network

4.3.1 Network Modelling

Another network used to evaluate the algorithm proposed in this work is an adaptation of a system proposed by Walski *et al.* [30]. This network is composed of one reservoir, R, with 97.5 meters of elevation. One pumping station, with two identical pumps, that pump water to a variable level tank T, that has an elevation of 160 meters. The minimum and maximum levels of the tank T are 0.5 and 12 meters, respectively, and the initial level is 6 meters. Eight consumption nodes have associated three distinct patterns of consumption, D1, D2 and D3, represented in figure 4.28. The base demand of each node D varies between 4.54 and 35.20 m³/h. The energy costs considered in this network are represented in figure 4.27 in which the tariffs for each hour are shown. Both pumps are similar, with the characteristic curve of the pump presented in figure 4.29. Other examples of curves of relative speeds considered in this work ($\omega = 0.5$ and $\omega = 2$) are included.

Initial values of the network modelling are presented in table 4.11. The total cost, for a simulation with a duration of 24 periods of 1 hour each, was shown to be of 499.42€.

Table 4.11: Initial values of energy and cost for the Walski Network

Pump	Energy Consumption (kWh/m ³)	Average Power (kW)	Maximum Power (kW)	Daily Cost (€) (€)
P1	0.26	22.75	29.86	249.71
P2	0.26	22.75	29.86	249.71
Total Cost				499.42

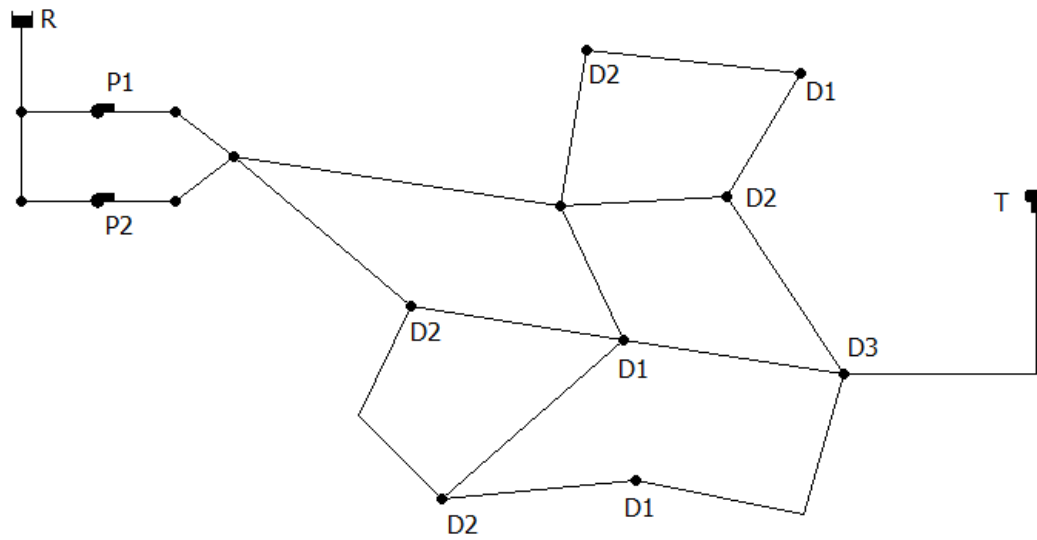


Figure 4.26: Scheme of the Walski network, drawn with EPANET software.

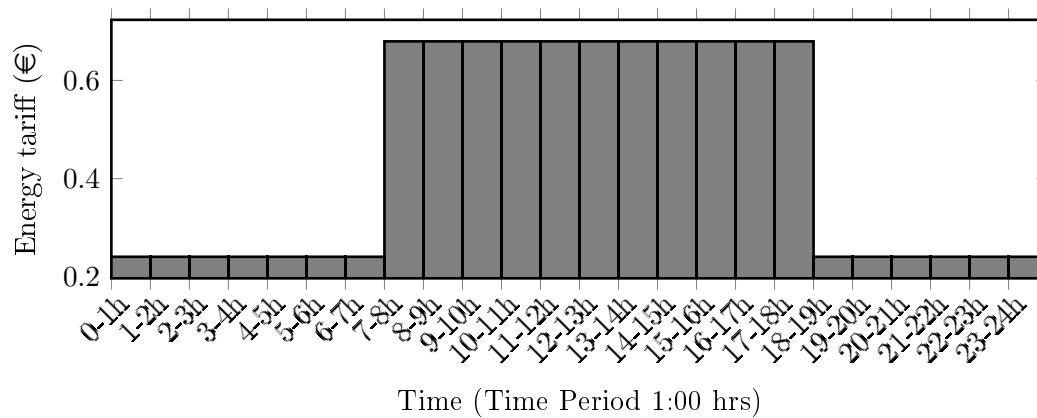


Figure 4.27: Energy tariff (€) associated with Walski Network.

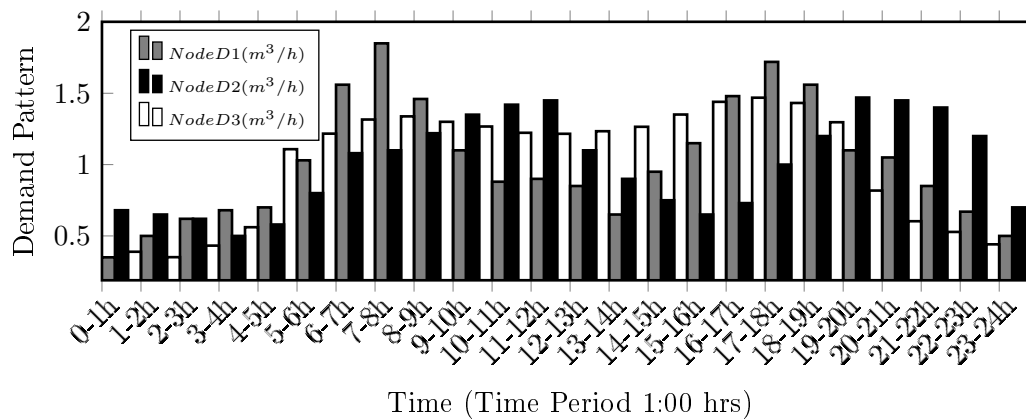


Figure 4.28: Consumption patterns associated with Walski Network.

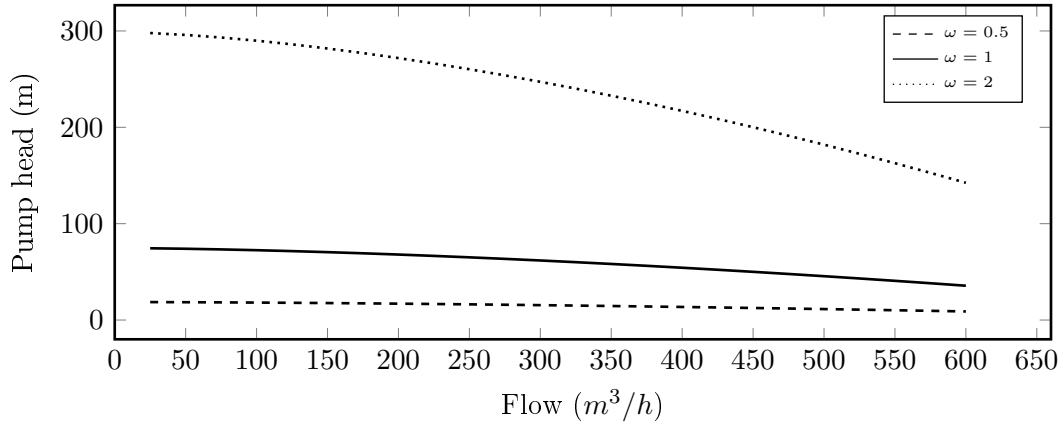


Figure 4.29: Characteristic curve of the pump used in the Walski Network problem, with the representation for three distinct relative speeds: the nominal speed ($\omega = 1$), the minimum speed ($\omega = 0.5$) and the maximum speed ($\omega = 2$)

4.3.2 Results Comparison

The optimisation problem for the Walski network can be represented by:

$$\begin{aligned}
 &\text{minimize} && f(\mathbf{x}) = \text{energy cost}, \\
 &\text{subject to} && h(\mathbf{x}) = \text{continuity condition} = 0, \\
 & && g(\mathbf{x}) = \text{levels limits} \leq 0, \\
 & && \mathbf{x}_i^{\min} < \mathbf{x}_i < \mathbf{x}_i^{\max},
 \end{aligned} \tag{4.14}$$

where $f(x)$ is the objective function, calculated using EPANET. Having two separated pumps, the number of variables is $i = 96$. Equation $h(x)$ are equality constraints and $g(x)$ are inequality constraints. for the pump activation time variables the values between 0 and 1 obtained from the optimisation can be transformed to minutes following a linear equation:

$$\text{time} = x_i \times 60.$$

For the variables of pump speed, the values obtained from the optimisation can be transformed to the relative speed of the pump ω following a linear equation:

$$\omega = 0.5 + (x_i \times 1.5).$$

The optimisation problem is subjected to the following equality constraint:

$$h(x) = L_{final} - L_{initial} = 0, \tag{4.15}$$

with $L_{initial}$ being the initial water level and L_{final} the final water level of the tank.

The optimisation problem is subjected to the following inequality constraint:

$$g1(x) = L - L_{max}, \tag{4.16}$$

$$g2(x) = L - L_{min}, \tag{4.17}$$

with L being the current water level, $L_{max} = 12$ being the maximum admitted level and $L_{min} = 0.5$ the minimum admitted level for the tank.

As with the optimisation of the Basic network, the ε DE method was used with a population of 50 individuals, a scaling factor of 0.7, a crossover rate of 0.9 and a maximum number of generations of 500 generations.

To solve the optimisation problem for the Walski network the L-BFGS-B method was given a disturbance value of 0.0001. The value given to the factor associated with the stopping criteria was $1.0e^7$.

In table 4.12 are presented the results of the optimisation of the Walski network presented in subsection 4.3.1 by the ε DE method, the DE method and the optimisation by the L-BFGS-B method, with a comparison of number of evaluation, CPU time, cost and cost reduction between the mentioned methods.

Table 4.12: optimisation results of Walski network benchmark for the DE method, the ε DE method and the L-BFGS-B method

Method	Evaluations	CPU time (min)	Cost €/day	Cost reduction (%)
ε DE	25000	50	342.1	31.5
		Pump 1	141.2	43.5
		Pump 2	200.9	19.5
DE	25000	40	312.0	37.5
		Pump 1	129.6	48.1
		Pump 2	182.4	26.9
L-BFGS-B	100	5	378.3	24.3
		Pump 1	189.1	24.3
		Pump 2	189.1	24.3

While the ε DE and the DE methods achieved less significant cost reductions than those obtained with Basic network, those reductions still range values from 30% to 40%. The best results were obtained by the L-BFGS-B method, although at the expense of constraint penalties. The low penalty coefficients given to the L-BFGS-B optimisation caused the continuity constraint to be violated. The results obtained from ε DE method are limited by the constraints previously presented, since the obtained solution does not incur in violations, and for that only solutions that have no error are considered by the method. The L-BFGS-B method has a lower reduction than the other two methods, but is much faster to reach it's optimum value. The evolution of the objective function for the L-BFGS-B method is compared to the evolution of the objective function for the ε DE method in figure 4.30.

The final solution obtained from the ε DE respects all constraints. Such situation can be confirmed from figure 4.31. Both minimum and maximum levels of the tank are respected, and the continuity equations is verified as well. The flows at the pump necessary for this solution are also present in figure 4.31. In figure 4.32 it is observable the controls given to the pumps after the optimisation. It is observable that the majority of the pumping is made in the morning period. While the majority of the time-steps the pumps operate at the nominal speed, exist a few exceptions. The ε DE method has difficulty finding feasible pump speeds, as the needed pump head needs to be guaranteed at all times. Overall, the reduction is greater at pump number 1, as is explicit in table 4.12.

The final solution obtained from the L-BFGS-B respects the tank level constraints.

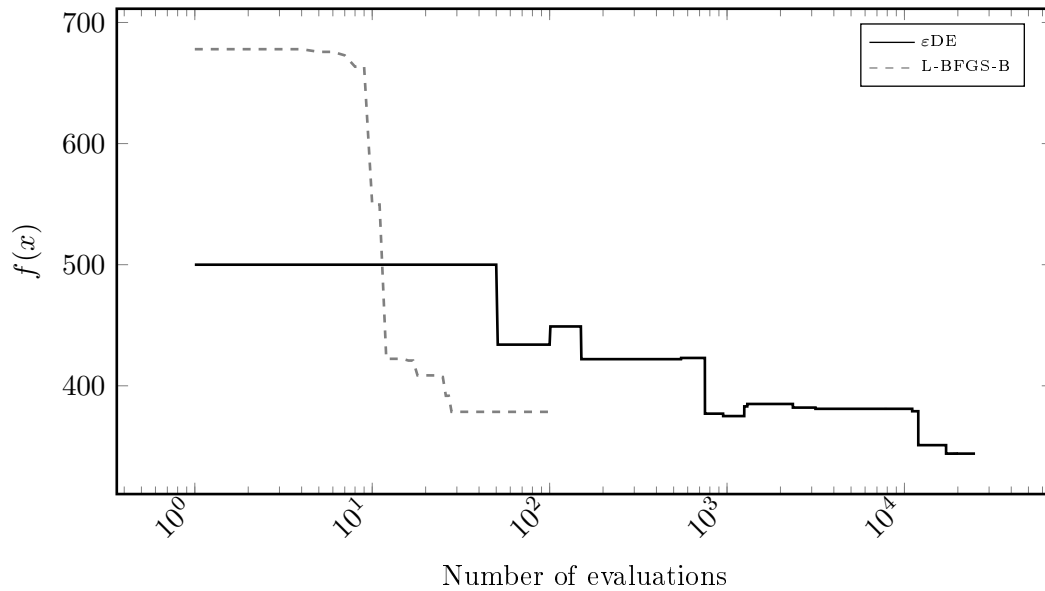


Figure 4.30: Cost function evolution throughout the optimisation process for Walski Network using the ϵ DE method (in black) and the L-BFGS-B method (in gray, dashed).

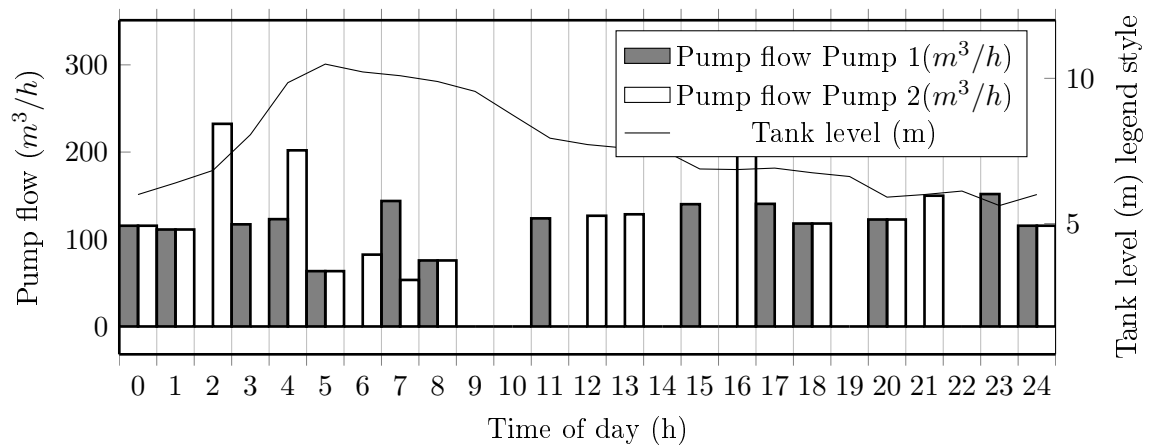


Figure 4.31: Pump flow (l/s) and tank level variation (m) during a day, after optimisation by the ϵ DE method.

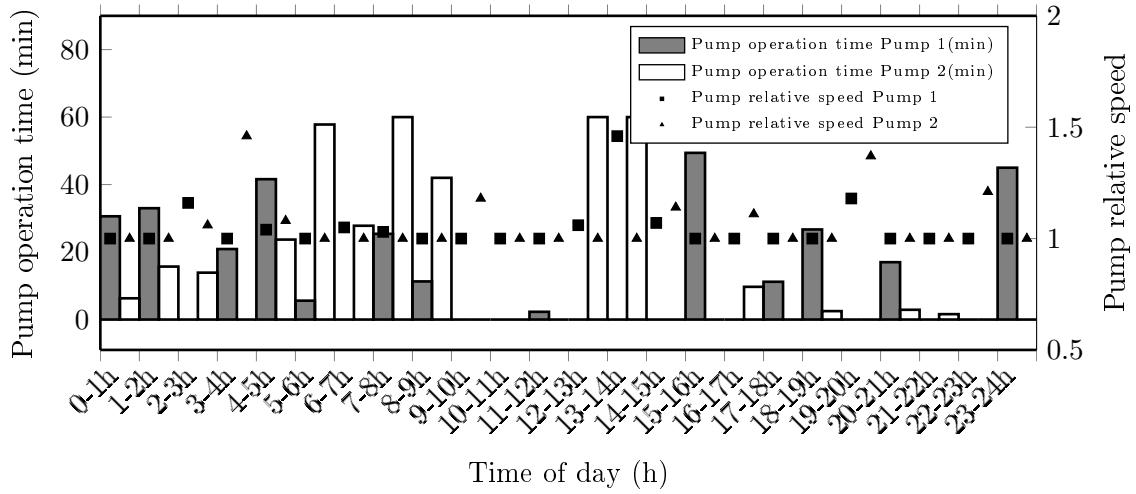


Figure 4.32: Pump controls (pump speed and operation time) during the day, after optimisation by the ε DE method.

Although the continuity level is not guaranteed. Such situation can be confirmed from figure 4.33. The flows at the pump necessary for this solution are also present in figure 4.33. In figure 4.34 it is observable that the pumping is made during the full day. With the L-BFGS-B method, the solution obtained is based in a small reduction of the pump relative speed, and a small reduction of the pumps operating time. As can be seen in figure 4.34 both pumps work at the same speed at all times, so the cost reduction in both pumps is the same, as can be seen in table 4.12.

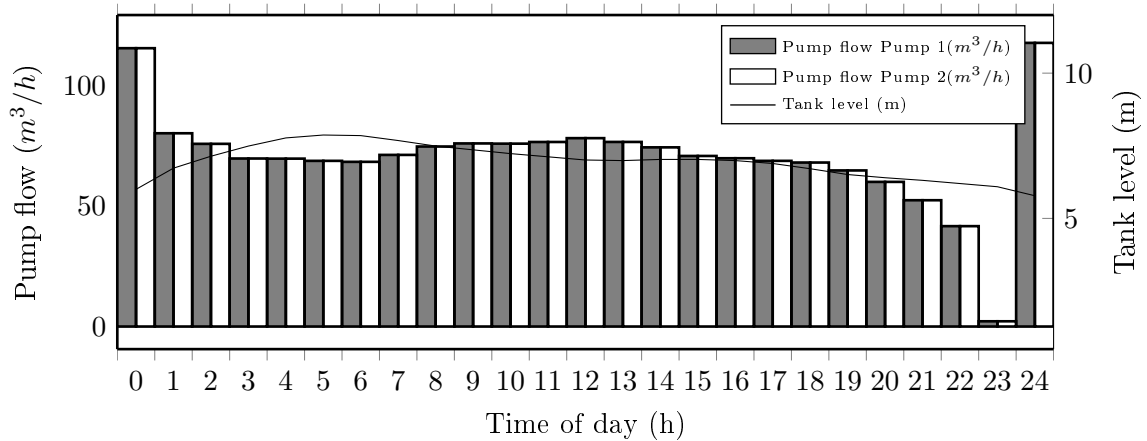


Figure 4.33: Pump flow (l/s) and tank level variation (m) during a day, after optimisation by the L-BFGS-B method.

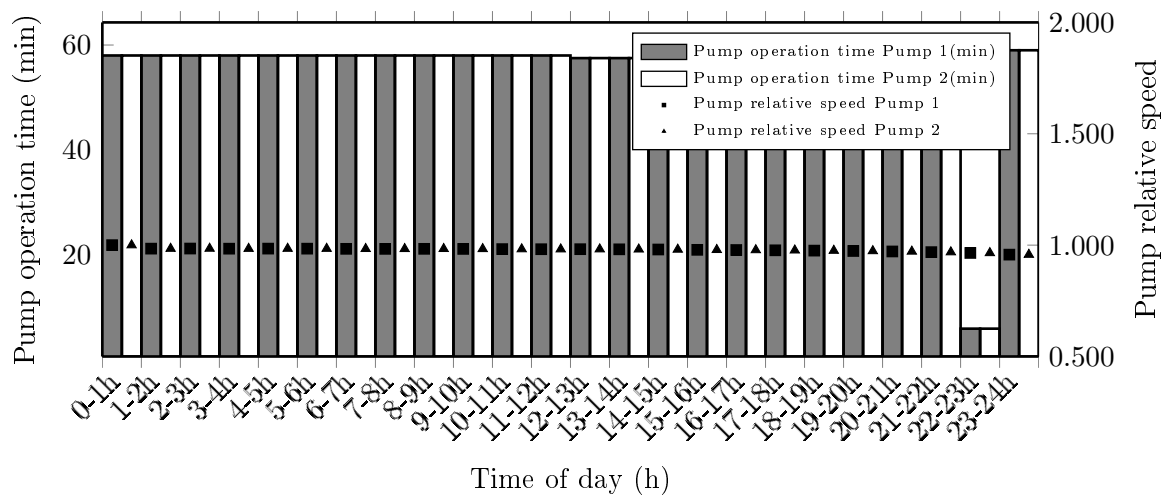


Figure 4.34: Pump controls (pump speed and operation time) during the day, after optimisation by the L-BFGS-B method.

Chapter 5

Final Remarks

5.1 Conclusions

This work focuses on the application of mathematical methodologies to reduce water pumping costs in Water Supply System, improving efficiency of pump controls. To this effect, two different optimisation algorithms were chosen to be used on the formulated problem. The ε Constrained Differential Evolution (ε DE), a modern algorithm from the field of Evolutionary Computation, and the Limited Memory Algorithm for Bound Constrained optimisation (L-BFGS-B), a classical algorithm based on the gradient of the formulated problem. The objective was to compare two different optimisation methods to evaluate performance of each one in this type of applied optimisation. These optimisation methods were tested on a variety of benchmark mathematical functions, to sort their feasibility in known conditions. Although these mathematical benchmarks were unconstrained functions, these tests allowed to sort the capability of these optimisation methods in complex functions. After these assessments, two different benchmark networks were tested, using the software EPANET to simulate the behaviour of the networks. First, a simple network, allowing to assess the work of the algorithm with WSS optimisation. After this, a more complex network was used so the number of optimisation variables were increased, which were directly connected to the number of pumps of the network. In the Basic network, which has one pump, the number of optimisation variables is 22 (initially 48). The more complex network has two similar pumps, that work separately, elevating the number of optimisation variables to 96. Both optimisation methods were compared to the results obtained from the optimisation using a DE algorithm. In the Basic network the cost reductions obtained were of 77.2% for the ε DE method and 79.7% for the L-BFGS-B method. The previously tested DE obtained cost reductions of 80.3%. In the Walski network the cost reductions obtained were of 31.5% for the ε DE method and 24.3% for the L-BFGS-B method. The previously tested DE algorithm obtained cost reductions of 37.5%. The results allow to conclude that the DE algorithm is the one able to get higher cost reductions. However, for it's nature of comparing errors first than cost, the ε DE algorithm would be more indicated to situations where constraints can't be violated. While the L-BFGS-B algorithm shows great qualities in terms of the speed of the evaluation in the Basic network evaluation, the results obtained for the Walski network are far from the expected results. Although the good results with the Basic network, the adding of Demand charges to the cost function, whose were non-existent in the Basic network, altered the nature of the cost function in the Walski network prob-

lem, creating a non-continuous problem. This situation hindered the capability of the L-BFGS-B method, not allowing the method to achieve results in the expected range. The design efforts on the GUI development produced a user-friendly interface of simple but functional look.

5.2 Future work

While part of a bigger project, this work intended to test two different algorithms in order to evaluate the best option to be used in the final optimisation software. And while the results obtained with the simulation from EPANET are encouraging, these optimisation methods should be put to the test with real Water Supply System. At the optimisation level, the use of different algorithms can be tested. Also, the use of two algorithms sequentially, with a modern algorithm, which are better algorithms at finding global minimums in situations with a complex solution space, followed by a gradient algorithm to refine the search. Although it can be time consuming, it could be beneficial if the costs would be further reduced. The HMI developed in this work should be adapted or used in the global software to be developed for the global project this thesis is included on.

Bibliography

- [1] P. Gleick, L. Allen, and M. Cohen, *The World's Water Volume 7: The Biennial Report on Freshwater Resources*. The World's Water, Island Press, 2011.
- [2] unknow author, "Portal da água." <http://portaldaagua.inag.pt/PT/InfoUtilizador/UsoEficiente/Pages/ConsumoPortugal.aspx>. Accessed: 10/03/2013.
- [3] J. van Zyl, D. Savic, and G. Walters, "Operational optimization of water distribution systems using a hybrid genetic algorithm," *Journal of Water Resources Planning and Management*, vol. 130, no. 2, pp. 160–170, 2004.
- [4] A. Bagirov, A. Barton, H. Mala-Jetmarova, A. A. Nuaimat, S. Ahmed, N. Sultanova, and J. Yearwood, "An algorithm for minimization of pumping costs in water distribution systems using a novel approach to pump scheduling," *Mathematical and Computer Modelling*, vol. 57, no. 3-4, pp. 873 – 886, 2013.
- [5] R. Hooke and T. A. Jeeves, "'direct search' solution of numerical and statistical problems," *J. ACM*, vol. 8, pp. 212–229, Apr. 1961.
- [6] J.-Y. Wang, T.-P. Chang, and J.-S. Chen, "An enhanced genetic algorithm for bi-objective pump scheduling in water supply," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10249 – 10258, 2009.
- [7] X. Zhuan and X. Xia, "Optimal operation scheduling of a pumping station with multiple pumps," *Applied Energy*, vol. 104, no. 0, pp. 250 – 257, 2013.
- [8] T. M. Walski, D. V. Chase, D. A. Savic, 1960, and I. Haestad Methods, *Water distribution modeling*. Waterbury, CT :: Haestad Press, 1st ed., reprinted with corrections june, 2001. ed., c2001. Web page: www.haestadmethods.com.
- [9] S. Ólafsson, "Chapter 21 metaheuristics," in *Simulation* (S. G. Henderson and B. L. Nelson, eds.), vol. 13 of *Handbooks in Operations Research and Management Science*, pp. 633 – 654, Elsevier, 2006.
- [10] S. Luke, *Essentials of Metaheuristics*. Lulu, 2009. Available for free at [http://cs.gmu.edu/~sim\\$sean/book/metaheuristics/](http://cs.gmu.edu/~sim$sean/book/metaheuristics/).
- [11] J. Brownlee, *Clever Algorithms: Nature-Inspired Programming Recipes*. Lulu Enterprises, 2011.
- [12] E. S. Raymond, "The art of unix usability." <http://www.catb.org/~esr/writings/taouu/html/ch02.html>. Accessed: 16/03/2013.

- [13] unknow author, “Mousesite.” <http://sloan.stanford.edu/mousesite/>. Accessed: 18/03/2013.
- [14] unknow author, “Parc.” <http://www.parc.com/about/>. Accessed: 16/03/2013.
- [15] <http://windows.microsoft.com/en-in/windows/history>. Accessed: 28/03/2013.
- [16] *GIMP User Manual*. The GIMP Documentation Team, 2000/2001.
- [17] unknow author, “The gtk+ project.” <http://qt-project.org/>. Accessed: 26/03/2013.
- [18] unknow author, “The gnome project.” <https://developer.gnome.org/gtk-faq/stable/x90.html>. Accessed: 26/03/2013.
- [19] unknow author, “Qt digia.” <http://qt.digia.com/About-us/>. Accessed: 26/03/2013.
- [20] unknow author, “Qt project.” <http://www.gtk.org/>. Accessed: 26/03/2013.
- [21] unknow author, “wxwidgets project.” <http://www.wxwidgets.org/about/>. Accessed: 26/03/2013.
- [22] unknow author, “wxwidgets license.” <http://www.wxwidgets.org/about/newlicen.htm>. Accessed: 26/03/2013.
- [23] D. Fadeyev, “Usabilitypost.” <http://www.usabilitypost.com/2009/04/15/8-characteristics-of-successful-user-interfaces>. Published: 15/04/2009. Accessed: 22/03/2013.
- [24] L. A. Rossman, *Epanet 2 users manual*. EPA-Environmental Protection Agency, September 2000.
- [25] E. Todini and S. Pilati, “Computer applications in water supply: vol. 1—systems analysis and simulation,” ch. A gradient algorithm for the analysis of pipe networks, pp. 1–20, Taunton, UK, UK: Research Studies Press Ltd., 1988.
- [26] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, “A limited memory algorithm for bound constrained optimization,” *SIAM Journal on Scientific Computing*, vol. 16, pp. 1190–1208, 1994.
- [27] T. Takahama and S. Sakai, “Fast And Stable Constrained Optimization by the Epsilon Constrained Differential Evolution,” *Pacific Journal of Optimization*, vol. 5, pp. 261–282, May 2009.
- [28] K. A. De Jong, *An analysis of the behavior of a class of genetic adaptive systems*. PhD thesis, Ann Arbor, MI, USA, 1975. AAI7609381.
- [29] B. Coelho, A. Tavares, and A. Andrade-Campos, “Analysis of diverse optimization algorithms for pump scheduling in water supply systems,” *EngOpt 2012 - 3rd International Conference on Engineering Optimization, Rio de Janeiro, Brazil*, 2012.
- [30] T. Walski and I. Haestad Methods, *Advanced water distribution modeling and management*. No. vol. 1, Haestead Press, 2003.