UNIVERSIDADE DE ÉVORA

ESCOLA DE CIÊNCIA E TECNOLOGIA

DEPARTAMENTO DE FISICA

Evolutionary Algorithms for the Modeling of Bioreactors

Sérgio Cavaleiro Costa

Orientação:

Professor Doutor Fernando Janeiro

Professora Doutora Isabel Malico

Mestrado em Engenharia Mecatrónica

Dissertação

Évora, 2016

UNIVERSIDADE DE ÉVORA

ESCOLA DE CIÊNCIA E TECNOLOGIA

DEPARTAMENTO DE FISICA

Evolutionary Algorithms for the Modeling of Bioreactors

Sérgio Cavaleiro Costa

Orientação:

Professor Doutor Fernando Janeiro

Professora Doutora Isabel Malico

Mestrado em Engenharia Mecatrónica

Dissertação

Évora, 2016

Dedicado aos meus pais e namorada!

"Premature optimization is the root of all evil."

Donald Knuth, 1974

"All models are wrong, but some are useful."

George E.P. Bose, Statistician, 1987

"The days when researchers emphasized using deterministic search techniques to find optimal solutions are gone."

Onwubolu and Babu, 2004

Acknowledgements

I want to thank my parents for all the support and opportunities they gave me.
To my girlfriend, Telma, who accompanied me throughout the academic course, supported me and gave me a lot of attention in the most complicated moments along the way. To her mother who also always encouraged me and supported me to increase my academic degree.
To my mentors (Professor Fernando Janeiro e Professora Isabel Malico) who guided me through this process, they gave me the opportunity to be inventive, supported me and made me feel integrated into a work and research team.
I still leave a word of appreciation to my colleagues in general, because without them it would not be easy either.

List of Symbols

N General nutrient

c Compound

X Biomass

ν Stoichiometric reaction coefficients

Y_i Biochemical reaction

M_i Molar Mass of i's compound

σ_i COD equivalent of the X compound

μ Growth specific rate

k Proportionality constant

r Biological kinetic reaction volumetric rate

x Cell concentration in reproductive state

$\vec{\theta}$ Vector of parameters being estimated

$J(\vec{\theta})$ Cost function

$y(\vec{\theta}, t)$ Set of output variables dependent on estimated parameters $\vec{\theta}$ and time t

$y^{\text{target}}(t)$ Set of reference output variables used for cost function $J(\vec{\theta})$ evaluation (from simulated or experimental data)

n GA population dimension

m GA number of allels

n_l NGA subpopulation dimension

$\vec{s_l}$ Vector of GA's subject positions for the NGA procedure

Acronyms

TPES	**T**otal **P**rimary **E**nergy **S**upply
GG	**G**reenhouse **G**ases
COD	**C**hemical **O**xigen **D**emand
ODE	**O**rdinary **D**ifferencial **E**quation
AD	**A**naerobic **D**igestion
VFA	**V**olatile **F**atty **A**cids
EA	**E**volutionary **A**lgorithms
GA	**G**enetic **A**lgorithm
NGA	**N**eighbour **G**enetic **A**lgorithm
IOP	**I**nverse **O**ptimization **P**roblem
OP	**O**ptimization **P**roblem
NLP	**N**on-**L**inear **P**rograming
MINLP	**M**ixed-**I**nteger **N**on-**L**inear **P**rograming

Evolutionary Algorithms for Bioreactors Modeling

Abstract

This work aims to study the application of Genetic Algorithms in anaerobic digestion modeling, in particular when using dynamical models. Along the work, different types of bioreactors are shown, such as batch, semi-batch and continuous, as well as their mathematical modeling.
The work intendeds to estimate the parameter values of two biological reaction model. For that, simulated results, where only one output variable, the produced biogas, is known, are fitted to the model results. For this reason, the problems associated with reverse optimization are studied, using some graphics that provide clues to the sensitivity and identifiability associated with the problem. Particular solutions obtained by the identifiability analysis using GENSSI and DAISY softwares are also presented.
Finally, the optimization is performed using genetic algorithms. During this optimization the need to improve the convergence of genetic algorithms was felt. This need has led to the development of an adaptation of the genetic algorithms, which we called Neighbored Genetic Algorithms (NGA_1 and NGA_2). In order to understand if this new approach overcomes the Basic Genetic Algorithms (BGA) and achieves the proposed goals, a study of 100 full optimization runs for each situation was further developed. Results show that NGA_1 and NGA_2 are statistically better than BGA. However, because it was not possible to obtain consistent results, the Nealder-Mead method was used, where the initial guesses were the estimated results from GA.

Keywords: Biogas, Bioreactor, Bioenergy, Inverse Methods, Optimization, Genetic Algorithms

Algoritmos Evolucionários para a Modelação de Bioreactores

Resumo

Neste trabalho procura-se estudar os algoritmos genéticos com aplicação na modelação da digestão anaeróbia e, em particular, quando se utilizam modelos dinâmicos. Ao longo do mesmo, são apresentados diferentes tipos de bioreactores, como os *batch*, *semi-batch* e *contínuos*, bem como a modelação matemática dos mesmos.

Neste trabalho procurou-se estimar o valor dos parâmetros que constam num modelo de digestão anaeróbia para o ajustar a uma situação simulada onde apenas se conhece uma variável de *output*, o biogas produzido.

São ainda estudados os problemas associados à optimização inversa com recurso a alguns gráficos que fornecem pistas sobre a sensibilidade e identifiacabilidade associadas ao problema da modelação da digestão anaeróbia. São ainda apresentadas soluções particulares de idenficabilidade obtidas através dos *softwares* GENSSI e DAISY.

Finalmente é realizada a optimização do modelo com recurso aos algoritmos genéticos. No decorrer dessa optimização sentiu-se a necessidade de melhorar a convergência e, portanto, desenvolveu-se ainda uma adaptação dos algoritmos genéticos a que se deu o nome de *Neighboured Genetic Algorithms* (NGA_1 e NGA_2). No sentido de se compreender se as adaptações permitiam superar os algoritmos genéticos básicos e atingir as metas propostas, foi ainda desenvolvido um estudo em que o processo de optimização foi realizado 100 vezes para cada um dos métodos, o que permitiu concluir, estatisticamente, que os BGA foram superados pelos NGA_1 e NGA_2.

Ainda assim, porque não foi possivel obter consistência nos resultados, foi usado o método de Nealder-Mead utilizado como estimativa inicial os resultados obtidos pelos algoritmos genéticos.

Keywords: Biogás, Bioreactor, Bioenergia, Métodos Inversos, Otimização, Algoritmos Genéticos

Contents

List of Figures

List of Tables

Chapter 1

Introduction

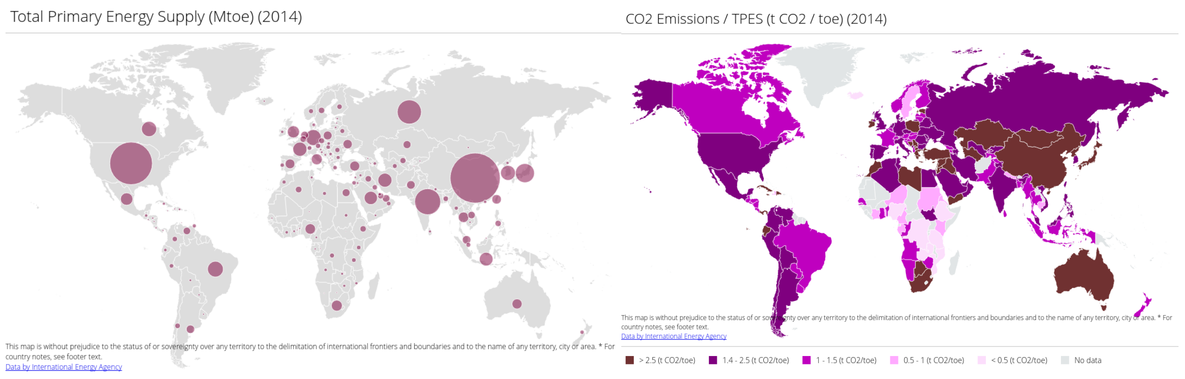
Biogas is a mixture of gases (mainly CO_2 and CH_4) produced by anaerobic digestion (AD) through the decomposition of organic matter. The two main sources of biogas for energy production are anaerobic digestion plants and landfill sites that receive significant amounts of waste each year. Biogas produced by AD plants is less complex and has a higher methane content than that generated in landfills and the use of AD plants is therefore likely to increase. Biogas can be produced from almost all kinds of feedstock such as agricultural waste, manure, municipal solid waste, plant material, sewage, green waste, food waste or industrial effluents.

For the control of biogas plants, processes that occur must be modeled and the models used calibrated. This calibration is performed in systems of equations that are complex and hard to be optimized. For that reason, the use of Evolutionary Algorithms (EA) is an often used technique that has been shown to be able to establish good estimates for that procedure.

Genetic Algorithms (GA) is a particular case of EA that shares some of the same principles that are found in microorganisms. Moreover their flexibility, easy understanding and implementation has made them popular. GA will be the fundamental focus of this work.

1.1 Motivation

One of the measures of country development may be the consumed energy. Countries like China, the United States of America, India, Japan, among others, consumed, back in 2014, great amounts of energy (figure 1.1a).


Total Primary Energy Supply (Mtoe) (2014)
This map is without prejudice to the status of or sovereignty over any territory to the delimitation of international frontiers and boundaries and to the name of any territory, city or area. * For country notes, see footer text.
Data by International Energy Agency
CO2 Emissions / TPES (t CO2 / toe) (2014)
This map is without prejudice to the status of or sovereignty over any territory to the delimitation of international frontiers and boundaries and to the name of any territory, city or area. * For country notes, see footer text.
Data by International Energy Agency
> 2.5 (t CO2/toe)
1.4 - 2.5 (t CO2/toe)
1 - 1.5 (t CO2/toe)
0.5 - 1 (t CO2/toe)
< 0.5 (t CO2/toe)
No data

(a) Total primary energy supply. (b) CO_2 Emissions.

Figure 1.1: World panorama at 2014 of energy supply and CO_2 emission [1].

Moreover, the previous maps show a link between total primary energy supply (TPES) and CO_2 emissions. The combustion of fossil fuels, among others, emit CO_2 into the atmosphere. This is the primary greenhouse gas emitted through human activities, although methane is also an important contributor to GHG emissions and has a higher global warming potential.
According to figure 1.2 the countries with higher TPES and CO_2 emissions are the same with the greatest population density.

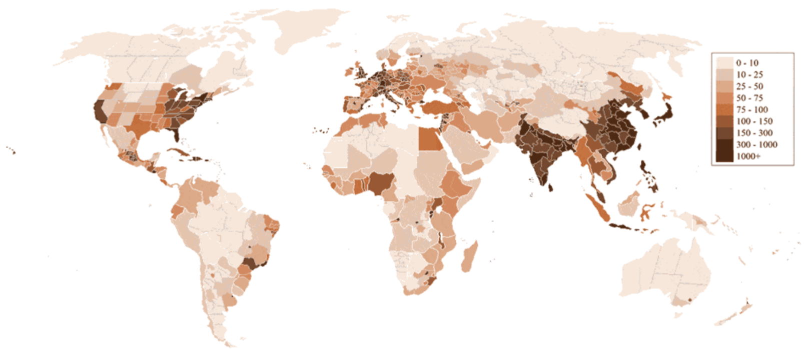


Figure 1.2: World population density (people/km^2) at the year of 2016 [2].

Since urban solid wastes and urban waste water, are some of the most important sources of biogas production, the most populous countries are good candidates for large biorefinery industry. Moreover, the use of biogas as a renewable energy source has many advantages. Among others, it helps decrease the dependency on fossil fuels, reduce the methane emissions to the atmosphere and is an answer to urban and industrial waste treatment [3].
Figure 1.3 presents some of the main applications for the biogas.

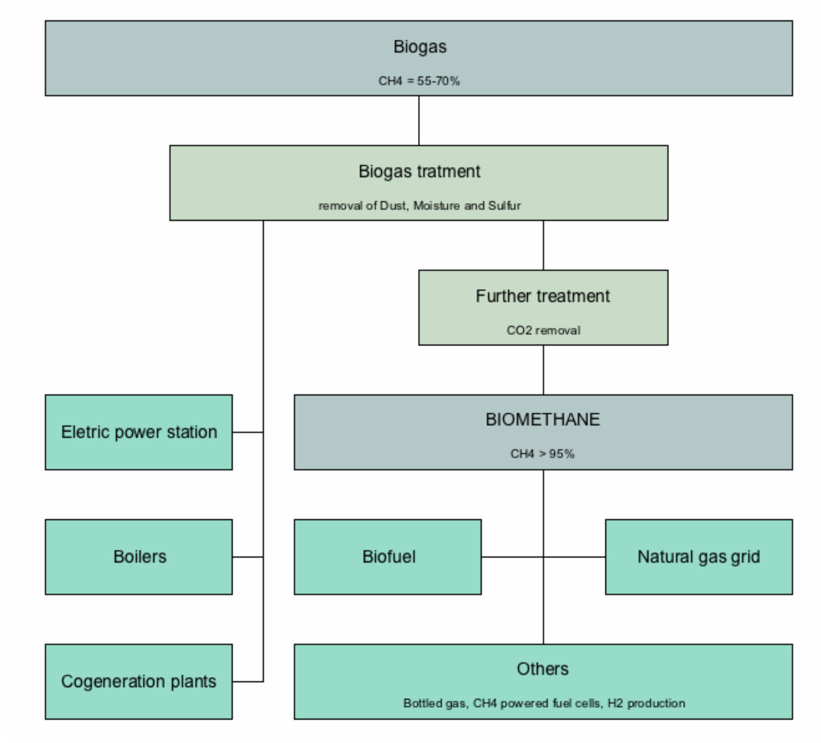


Figure 1.3: Biogas application (adapted from [4]).

Depending on biogas application, treatments may be required so that the bioproduct respects several required conditions. This is the case when injecting the biogas in a natural gas grid or when using it as for transports, biofuel and others. In those cases, biogas must reach a mix of at least 95% of CH_4. In that case the biogas is then called biomethane. Nevertheless, mixes with ranges between 55% and 70% of CH_4 can, with minor handling be used in electrical power stations, boilers and/or cogeneration plants.

According to figure 1.4, Europe has been leading the industrialization of biogas, representing 60% of the world's production, followed by the Asia Pacific Region.

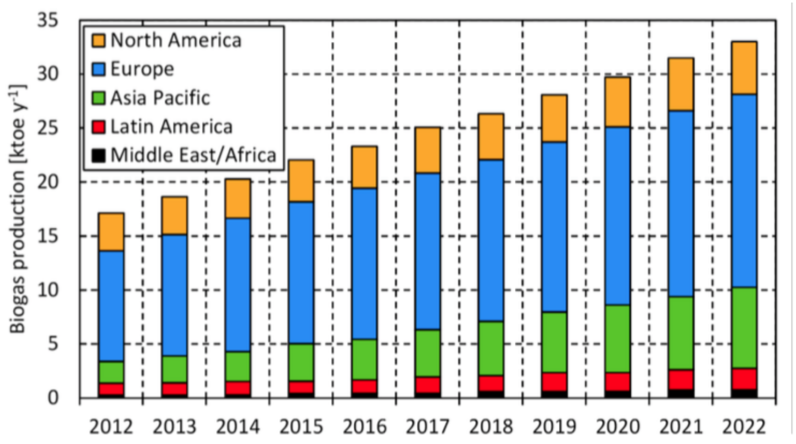
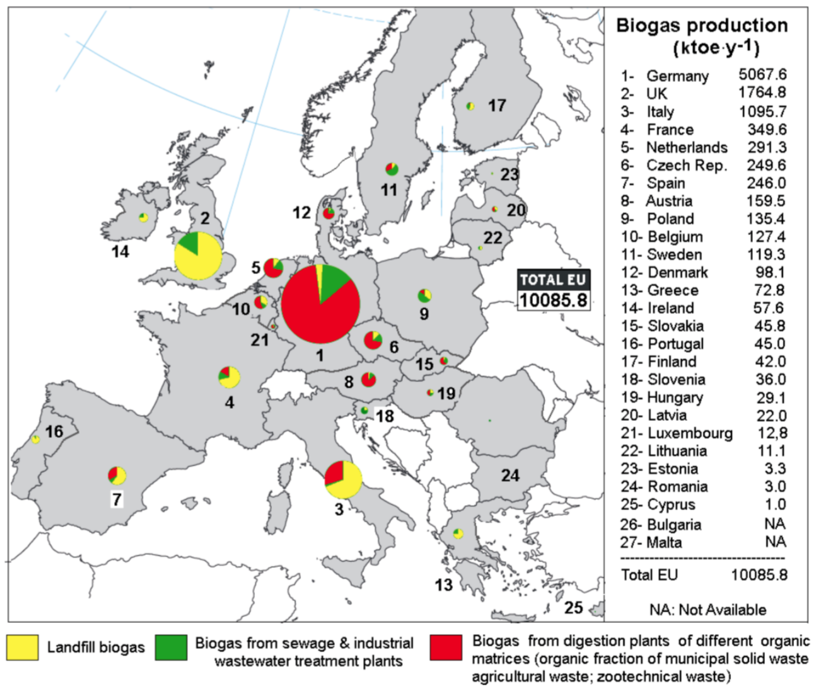


Figure 1.4: World biogas production by region from 2012 to 2014 and future prediction until 2022 [4]

One of the reasons for Europe's leading role, are the effective policies that have been implemented by north-central EU countries and that promote the use of biomethane for public and private transportations [4].

Figure 1.5 shows that, in Europe, Germany, the UK and Italy are the countries that are leading this industry. While the UK's and Italy's most common source is the landfills, Germany shows a more indiscriminate source.


TOTAL EU
10085.8
🟨 Landfill biogas
🟩 Biogas from sewage & industrial wastewater treatment plants
🟥 Biogas from digestion plants of different organic matrices (organic fraction of municipal solid waste agricultural waste; zootechnical waste)

Biogas production (ktoe·y-1)

1- Germany	5067.6
2- UK	1764.8
3- Italy	1095.7
4- France	349.6
5- Netherlands	291.3
6- Czech Rep.	249.6
7- Spain	246.0
8- Austria	159.5
9- Poland	135.4
10- Belgium	127.4
11- Sweden	119.3
12- Denmark	98.1
13- Greece	72.8
14- Ireland	57.6
15- Slovakia	45.8
16- Portugal	45.0
17- Finland	42.0
18- Slovenia	36.0
19- Hungary	29.1
20- Latvia	22.0
21- Luxembourg	12,8
22- Lithuania	11.1
23- Estonia	3.3
24- Romania	3.0
25- Cyprus	1.0
26- Bulgaria	NA
27- Malta	NA
Total EU	10085.8

NA: Not Available

Figure 1.5: Europe biogas production by countries and source [4].

There are still some drawbacks with the AD technology. One of the major ones is that the biogas that results from AD only has 55%-65% of CH_4 [4]. Moreover, some of the other components are CO_2, water vapor, traces of H_2S and H_2 and possibly other contaminants (such as siloxanes). Most of this remaining constituents are chemically aggressive for the direct use in the majority of the applications shown in 1.3 without any treatment. The problem is that when in the presence of H_2O, this element react with most metals and the reactivity is enhanced by concentration and pressure, by the presence of H_2O and at elevated temperature [4].

1.2 Objectives

To produce biogas technological requirements are needed. One of those is the existence of a control module capable of adapting to the different states present in the biorefinery plant. But, in first place a state mathematical model must be obtained in order to understand the best way to design that controller.

In this work, the Campestrini's et al. mathematical model [5] will be used. In this model there are only three stages for the biogas production, so the major phenomenas may be used without the complexity of others.

There model also includes multiple constants that are dependent on different conditions of substrate and bacterial population. In order for the mathematical model to be representative of a particular bioreactor running under specific conditions, the calibration of those parameters is required. In this work this calibration process is investigated and performed using Evolutionary Algorithms (EA), such as the Genetic Algorithms (GA).

During the development of this work, in the GA optimization process problems were found that led to the development of a new adaptation of the GA. To the new hybridization of the GA, it was called Neighbored Genetic Algorithms NGA, since it makes use of a subpopulation that searches locally for better parameter combinations in order to improve the convergence rate of GA.

Due to this new approach to the GA, a demonstration that the Basic Genetic Algorithms BGA is outperformed by the NGA is obtained. And since both are based in an heuristic approach this study is performed with statistical bases.

1.3 Structure

This work is divided into 4 major chapters. In chapter 2, an overview of bioreactors and the microorganism behavior, growth and death is performed. In the first section, the different types of bioreactors are explored, as well as their influence in the substrate concentration. This is followed by a section (section 2.2) where the foundations of microbial growth in a bioreactor context are presented. In this section, the mathematical relationship that models that growth is related with the substrate concentration, cell death, population saturation and inhibitors is presented.

Section 2.3 presents, among other models found in literature, the Campestrini et al. model, which is the onde chosen in this work for the optimization process. In the last section of this chapter (section 2.4), a simulation base is established, as a first approach for the Inverse Optimization Process, IOP.

Chapter 3 explains how optimization problems are subdivided, how to pose an optimization problem (in particular, an Inverse Optimization Problem, IOP) and some typical cost functions used for the type of IOP developed in this work. A review of possible cost functions found in literature is also performed. Moreover, to get more sensitivity to the incoming problems, plots of pairs of parameters versus the cost function value is also presented.

A description of the GA problems and how they can be evolved through hybridization, and, in particular, through the Memetic strategy is then presented. Finally a new twist to the GA method is proposed to get better performances (the NGA method).

In the last chapter the optimization problem with all parameters to be estimated is presented. In this chapter, BGA is also compared BGA with NGA. In an attempt of getting better results, a Nelder-Mead optimization algorithm is used with the GA fitted parameters as first guess.

1.4 Original contributions

The work developed in this thesis was presented at 2 conferences:

- S. Cavaleiro Costa; I. Malico; F. M. Janeiro; A. P. Baptista; A. Coelho; T. Lopes da Silva; I. P. Marques. Dynamic modeling of biogas production supported by an experimental anaerobic digestion process. 11th Conference on Sustainable Development of Energy, Water and Environmental Systems, Lisbon, Portugal, 2016.
- S. Cavaleiro Costa; F. M. Janeiro; I. Malico. On the use of genetic algorithms for parameter identification in anaerobic digestion modelling. 12th International Conference on Diffusion in Solids and Liquids: Mass Transfer, Heat Transfer and Microstructures and Properties, Split, Croatia, 2016.

The first publication presentes the development of a combined experimental and numerical approach to study the anaerobic digestion of both the wastes produced in a biorefinery using yeast for biodiesel production and the wastes generated in the preceding microbial biomass production. The experimental results show that it is possible to valorise through anaerobic digestion all the tested residues. In the implementation of the numerical model for anaerobic digestion, a procedure for the identification of its parameters needs to be developed. A hybrid search Genetic Algorithm was used, followed by a direct search method. In order to test the procedure for estimation of parameters, first noise-free data was considered and a critical analysis of the results obtain so far was undertaken. As a demonstration of its application, the procedure was applied to experimental data.

The second work presents the identifiability analysis of a nonlinear dynamical model for a batch reactor. Particular attention is given to the structural identifiability of the model, which considers the uniqueness of the estimated parameters. To perform this analysis, the GenSSI toolbox was used. The estimation of the model parameters is achieved with genetic algorithms which have already been used in the context of AD modelling, although not commonly. The work discusses its advantages and disadvantages.

Chapter 2

Anaerobic Digestion

Anaerobic digestion (AD) is a sequence of biological processes that occur where organic matter (carbohydrates, proteins, lipids or more complex compounds) is transformed into methane, carbon dioxide and anaerobic biomass, in an oxygen-free environment [6].

In section 2.1 the different types of bioreactors and mixtures is presented, in section 2.2 the mathematical formulation of microbial growth is presented as well as their relation with the substrate consumption and different condition influences. A review of mathematical models used in AD is presented in section 2.3 and in particular, section 2.4 solves and establish conditions for the Campestrini et al model.

2.1 Bioreactors

A bioreactor is a vessel in which biological reactions occur. This is a key element of any biotechnological process.
When it comes to biological reactors, or bioreactors, it is up to referring to a technology with broad spectrum ranges and applications. This can be used to obtain high-value-added products with applications in health, effluent treatment, among others. In bioreactors can be placed an inoculum, i.e., an amount of substance with enzymes, microorganisms, cultures or cells of higher organisms, so that, the complexity grows with the diversity of nutritional, physiological, metabolic and morphological characteristics. All these, must therefore, be controlled and integrated into the equipment so that it provide the optimal production conditions.
Bioreactors may be classified by their operation mode or by if they are mixed or not [7].
In a mode of operation sense, a bioreactor may be:

- Batch;
- Fed-batch;
- Continuous.

Figure 2.1 shows some generic examples of the different modes of bioreactor substrate concentration evolution in time.

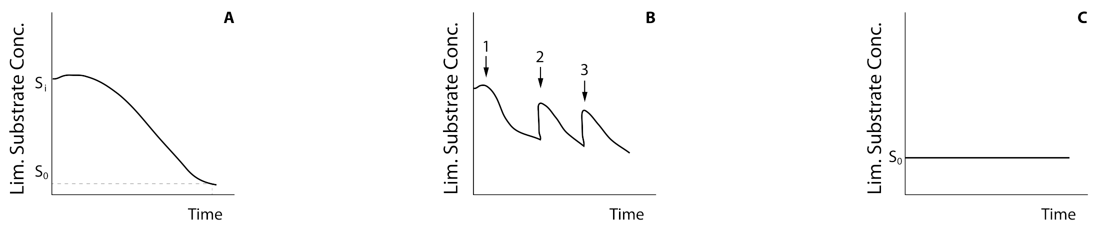
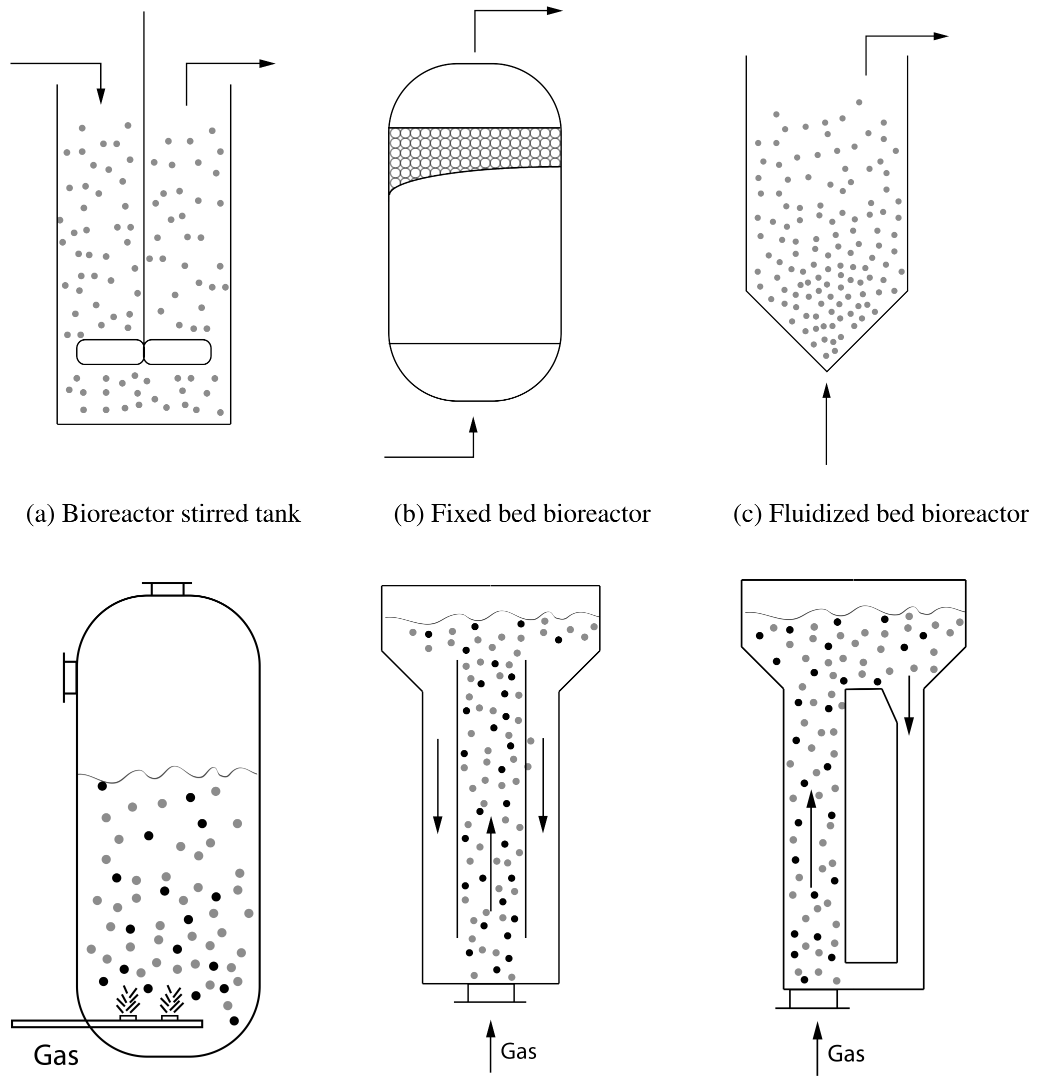


Figure 2.1: General operation of each mode; A-Batch; B-Fed-Batch; C-Continuous.

In batch operation all the inoculum is introduced only at the beginning. Therefore, concentration is not controlled being able to vary as the living cells take them up. Only at the end, the sub-products of the process are harvested. Nevertheless, there is some control of the process that can be taken such as: PH, temperature, dissolved oxygen, which are held constant, and foam.
Fed-batch reactors are those which have had greatest growth in the past three-decades [8]. They have been used in fermentation, biotechnology, chemical and waste treatment industries.
Both, batch and fed-batch reactors, are, usually, used in low-volume, high-value products such as fermentation products, including amino acids and antibiotics, recombinant DNA products, and special chemicals [8].
Continuous bioreactors can further be divided in [7]:

- Piston flux bioreactor
- Perfect mix bioreactor
- Multiple associated bioreactors
- Escalated fed bioreactor

Mixing is another important bioreactor characteristic, since the amount of gas present in the substrate may influence microorganisms growth. Figure 2.2 shows some examples of strategies typically used for the mixing.



(a) Bioreactor stirred tank (b) Fixed bed bioreactor (c) Fluidized bed bioreactor

(d) Bioreactor bubble column (e) Air-lift Bioreactor (f) Air-lift Bioreactor

Figure 2.2: Most common bioreactors configurations (adapted from [7]).

2.2 Microbial Growth Modeling

The Anaerobic Digestion (AD) is an extreme complex process and for that reason its control is an hard task. Bioreactors often break-down and completely stop biogas production. On the other hand, biogas is a growing option as a source of energy, specially in Europe. Moreover, it should be one of the most obvious sources since the necessary resources are some of the most accessible, such as agricultural waste, manure, municipal waste, plant material, sewage, green waste or food wastes. For these reasons there modeling is not only useful, but in the sequence of European directives, is an imperative technology with an increasing necessity of developing and comprehension.

An AD consists of a very complex process where environmental conditions play an important role. Moreover, it depends of activity coordination of multiple complex microbial associations to transform organic material into mostly CO_2 and methane, CH_4. Figure 2.3 shows a simplified scheme of the sequence of processes involved.

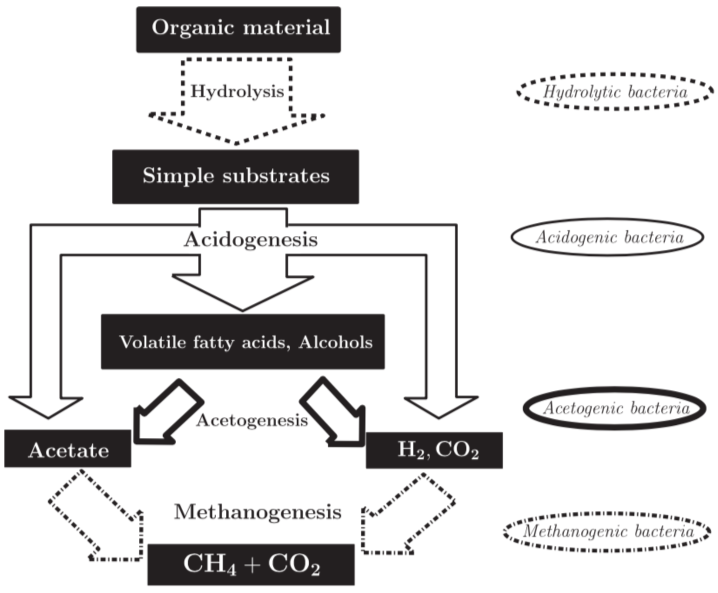


Figure 2.3: Anaerobic digestion scheme [9].

In the hydrolysis process insoluble organic material and high molecular weight compounds are degraded (as lipids, polysaccharides, proteins and nucleic acids) into soluble organic substances (such as amino acids and fatty acids) [10].

This simple substrates are further split during acidogenesis (by acidogenic bacteria, also known as fermentative) from where results Volatile Fatty Acids (VFA), ammonia (NH_3) , CO_2, H_2S and other by-products.

The next stage is performed by the acetogens bacterias, where the higher organic acids and alcohols are further digested to mainly produce acetic acids as well as CO_2 and H_2. This conversion is primarily controlled by the partial pressure of H_2 in the mixture.

The last stage is performed by two groups of methanogenic bacteria, one splits acetate into methane and carbon dioxide and the other one uses hydrogen as electron donor and carbon dioxide as acceptor to produce methane.

A biochemical reaction where the N nutrient is transformed in a series of c_i compounds and biomass X is expressed, in a general form as,

$$-\nu_n N + \sum_{i=1}^{n_c} (\nu_i c_i) + \nu_x X = 0 \tag{2.1}$$

where ν_n, ν_i and ν_x are the stoichiometric reaction coefficients. It is useful to normalize equation using the Chemical Oxygen Demand (COD), getting,

$$-N + \sum_{i=1}^{n_c} (Y_i c_i) + Y_x X = 0 \tag{2.2}$$

where Y_i are biochemical reaction yield that can be evaluated by,

$$Y_i = \frac{\nu_i}{\nu_n} \tag{2.3}$$

M_i is the molar mass of i's compound and σ_i is the COD equivalent of the X compound.
Following the same idea, M_x can be defined as the molar mass of the biomass and σ_x the equivalent biomass COD.
Moreover, in order to have mass conservation,

$$\sum_{i=1}^{n_c} Y_i + Y_n = 1 \tag{2.4}$$

In general, in biological kinetic reactions volumetric rates (r) of formation and consume are quantified with dimension given in $L^{-3}MT^{-1}$ and specific rates in T^{-1}. While the first are set in units of time and volume of liquid in the bioreactor, the latter refer to the mass formed or consumption per unit of time and biomass present in the reaction vessel. In general, specific rates are expressed as q excluding the growth specific rates that are usually expressed as μ.

The simplest and more often used models to express the microbial growth are the unstructured models which consider that cellular population is homogeneous both from the metabolic point of view and structurally. Because of this, cellular population growth is estimated using the number or the mass of cells only [7].

The unstructured model 2.5 shows a first order Ordinary Differential Equation (ODE) which describe the microbial growth without inhibition. This model was introduced by Malthus and represents a first order autocatalitic reaction.

$$\frac{dx}{dt} = r_x = \mu x \tag{2.5}$$

It relates the x cell concentration in reproductive state with r, the volumetric rate of biomass accumulation, by a proportion μ of specific growth rate in T^{-1}.

Integrating 2.5 between the initial time t_0 and t, is obtained,

$$x = x_0 e^{\mu(t-t_0)} \tag{2.6}$$

To express dependency between the specific growth rate and the substrate concentration, Monod, in 1950, developed the relation,

$$\mu = \mu_{\max} \frac{S}{K_S + S} \tag{2.7}$$

Where K_S is called the saturation constant. The higher the affinity of micro-organism with the substrate the lower the value of K_S. Note that growth rate approaches zero, if $K_S >> S$. On the other hand, if $K_S << S$, the growth rate equation is of first order and if $K_S \cong S$ than the order remains between 0 and 1.

In a more general situation, when there n_S limiting substrates may exist, equation (2.7) can be described by,

$$\mu = \mu_{\max} \prod_{i=1}^{n_S} \frac{S_i}{K_{Si} + S_i} \tag{2.8}$$

In figure 2.4 is shown the microorganisms growth as function of the substrate concentration.

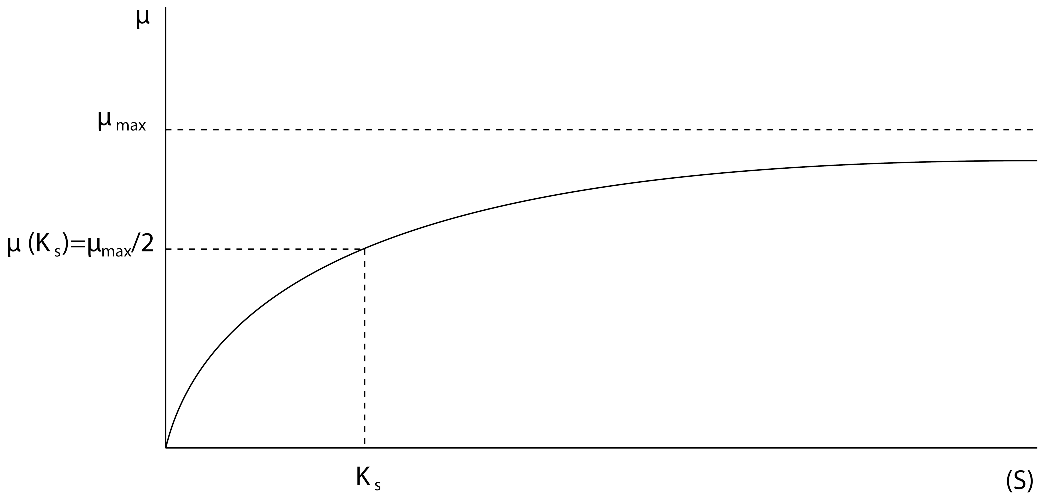


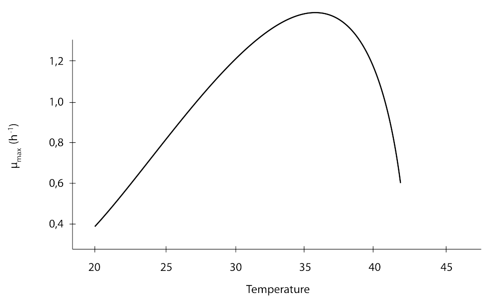
Figure 2.4: Microorganisms specific growth rate as a function of substrate concentration described by the Monod Law.

The substrate is a source of carbon, nitrogen and oxygen for microorganisms, but, since cells are fed with these compounds, once the substrate concentration reaches a limit, growth must start decreasing until stagnation is reached. This maximum is described by $\mu_{\max}$, moreover when μ reaches the value of K_S it should be half the the way to $\mu_{\max}$ 2.4.

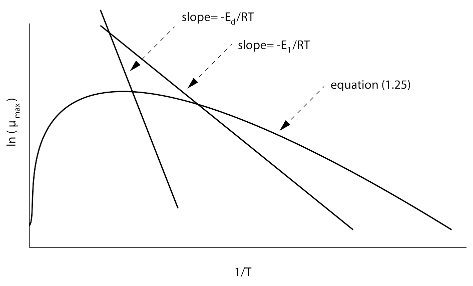
One of the limiting growth parameter, for example, is the temperature. In Hinshelwood's ([11]) work, is developed a relation between $\mu_{\max}$ and de temperature,

$$\mu_{\max}(T) = A_1 e^{-\frac{E_1}{RT}} - A_d e^{-\frac{E_d}{RT}} \tag{2.9}$$

where, A_1 and A_2 are some constants of the environment for a substrates composition and E_1 and E_d represents the activation for the thermal bacterial growth and death, respectively.
Figure 2.5 shows how the limit varies with the temperature and how the present parameters parameters shapes the curve.



(a)



(b)

Figure 2.5: Representation of how temperature influence the microorganisms growth limit. In (a) can be found the temperature influence in microorganisms growth while in (b) is shown the relationship between parameters and the evolution shape.

In particular, for a batch bioreactor type, figure 2.6 shows the stages observed in the evolution.

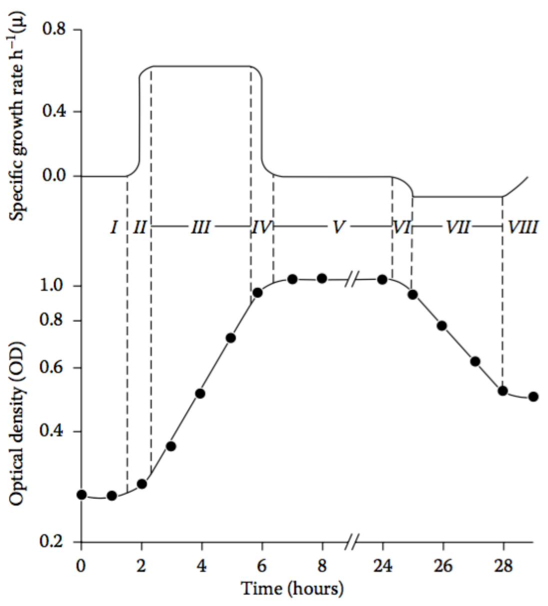


Figure 2.6: Stages for micro-organisms growth in batch cultures [12].

Figure 2.6 shows a typical pattern of the growth curve which is observed in a batch fermentation process, this is known as the *growth cycle* and is divided as:

I - Lag phase;	V - Stationary phase;
II - Acceleration phase;	VI - Accelerated death phase;
III - Exponential (logarithmic) phase;	VII - Exponential death phase;
IV - Deceleration phase;	VIII - Death or survival phase.

The lag phase is the period when cells are adapting to the substrate. In this period, organisms are confronted with the need to decompose carbon that is structured in the substrate in a particular way so they need to evolve and, in some cases, develop enzymes to achieve it.

In an industrial process, this phase is intended to be as short as possible since there is no production of biomass and considerable costs are involved.

To evaluate if a particular organism has entered in phase one it is just needed to plot a $\log n$ (biomass) versus time as shown in figure 2.7.

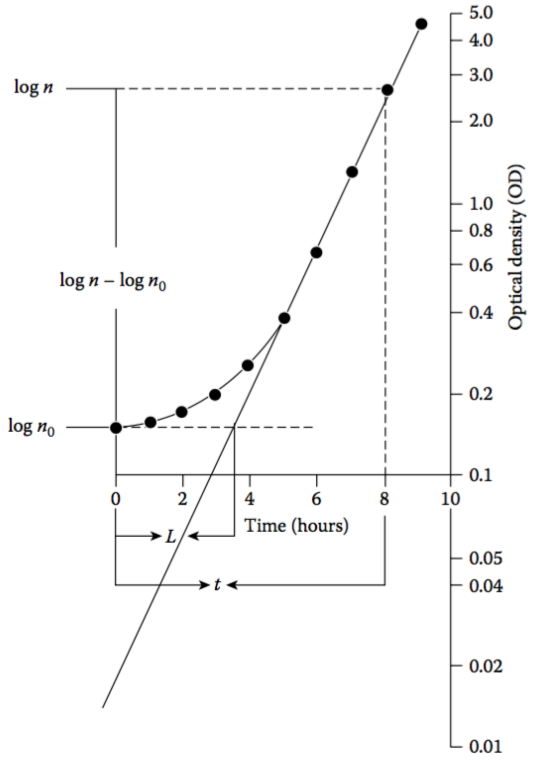


Figure 2.7: Graphical determination of the lag phase and the number of viable cells at the onset of batch fermentation [12].

From figure 2.7, The acceleration phase is easily identified by extrapolating the lag phase sideways and the exponential phase downward as shown, with the point of interception (L) taken as the time at which the lag phase ended.

If exponential phase continues to extrapolate downward, then the point at which the ordinate is intersected gives the number of cells that were viable and metabolically active at the point of inoculation.

To find where a lag (L) has occurred during the course of the fermentation process and for how long can be determined by,

$$\frac{\log n - \log n_0}{t - L} = \frac{\log 2}{T} \tag{2.10}$$

Where n is the total number of cells for an instant time t, n_0 is the cell's number at the beginning of the process and T is the organism's mean generation time (*doubling time*).

The exponential phase is the one where cells are comfortable with the substrate and are capable of transforming the primary carbon source into biosynthetic precursors, reducing power and energy. Figure 2.8 shows this phase.

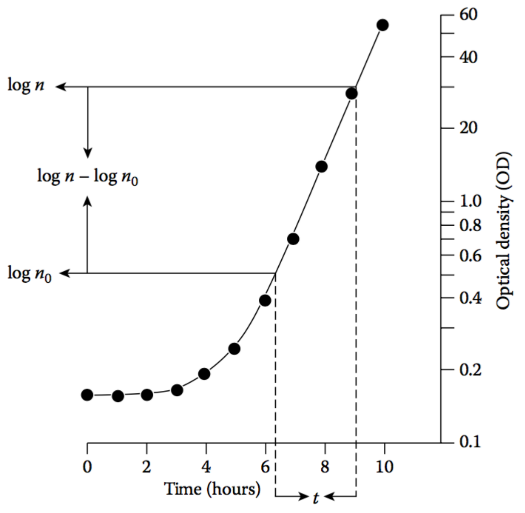


Figure 2.8: Exponential phase [12].

In this phase for a given inoculated cell's number, n_0 into a suitable medium, than after one generation there are $2n_0$ cells and so, after z generation there will be $2^z n_0$.

Another important phenomena that occurs during the process is the microbial death. They are important because this cells stays in the substrate and when lysis occur cells membrane ruptures and the cell contents spread in the environment contributing to the alterations of the substrate and influencing growth of the other cells.
To model this phenomenon, a first order equation can relate the rate at which death occurs r_d with the micro-organisms concentration,

$$\frac{dx_d}{dt} = r_d = k_d x \tag{2.11}$$

For the cell can be in an active state, there is an operation set that must be performed, just to stay alive. This processes are called the cells *maintenance*. Those are the processes that must exist for cells survival, but has no direct contribution for the generation of new cells nor the biosynthesis of

extracellular products.
This can be postulated as,

$$r_{S_m} = m_S x \tag{2.12}$$

In some cases, the subdivision of substrate components may be useful, for that, the volumetric consume rates of its components may be expressed as a sum,

$$r_S = \sum_{i=1}^{n_S} r_{S_i} \tag{2.13}$$

In a compact form, the mass balance of biomass nutrients and products, for a batch bio-reactor type, can than be summarized as,

$$\begin{cases} \dot{x} = (\mu - k_d)\, x \\ \dot{x}_d = k_d x \\ \dot{S} = -q_S x \\ \dot{c} = q_c x \end{cases} \tag{2.14}$$

2.3 AD Models

For the increasing necessity in AD modeling, multiples attempts may be found in literature, for example, L. Appels et al. [10] and Andres Donoso-Bravo [6], makes an extensive review of models typically used in the literature. Depending of there application, there selection may be an important criterion since models complexity may be extremely large, for example, among of the cited authors by L. Apples work is D. Batstone with the AMD1 model which makes use of 30 state variables.
Due to the experienced difficulties in having good models in a simple way, one alternative that has spread is the neuronal networks. This alternative, although has no information about how inside processes are occurring only concerning with input and output data. Moreover, it needs representative training data, which may arrive from experimentations or making use of models. In both cases this may represent an effort that will give no understand of the problem and have high costs associated with lower robustness.
Struggle to understand phenomenons that occur in AD has been modeled by multiple authors. It

is usual that if a Monod model is followed than substrate or microorganism information is needed. The problem is that evaluation of this experimental measurements is expensive and so, typically, only biogas is quantified. In those situations, although arrives identifiability problems [13]. In that work, Müller et al., has suggested a model using two steady state equations given by the VFA, and bacterias and the output equation, the biogas production. In that work is compared two growth rate models, one using the Monod Law and the other uses substrate inhibition. Due to biogas being the only observed variable, options taken in that works to get identifiability include assuming known parameters, direct measurement of initial conditions and excluding inhibition constrains. Beside that, it has been assumed boundaries based on experimental measurements from [14], this way is increased possibility of locally identifiability become achievable in that domain. In Lokshina's work [15], is shown difficulties in the half-saturation parameter, K_S, both increasing and decreasing at low temperature AD. Moreover, Monod and a generalized Haldane models are presented.
Haldane model is a Monod Law alternative where substrate inhibition may be included by the K_I parameter,

$$\mu = \mu_{max} \frac{S}{K_S + S + S\left(\frac{S}{K_I}\right)^n} \tag{2.15}$$

Where n is known as the Haldane index and for $n = 1$, Haldane model is said *Haldane 1* and for $n = 2$ is said *Haldane 2*. Moreover, note that, if the inhibition parameter rises than that term is brought to zero and Monod Law is obtained.
In [14] a model with three stages is presented where both Monod and Haldane models are included for acidogenic and methanogenic bacterias, respectively.
In [16] can be found further more models, such as Chen & Hashimoto kinetic model (2.17) and Contois kinetic model (2.16).

$$\mu = \mu_{max} \frac{S}{K_x x + S} \tag{2.16}$$

$$\mu = \mu_{max} \frac{S}{K S_0 + (1 - K)S} \tag{2.17}$$

Where, K_x in known as the Contois kinetic constant and K is the Chen and Hashimoto dimensionless kinetic constant.

In particular, for the purposes of this work the model used will be the one described in [17].
It considered that there are two biological reactions, the acidogenesis and the methanogenesis. In the first stage the chemical reaction is described by,

$$k_1S_1 \rightarrow x_1 + k_2S_2 + k_4CO_2 \tag{2.18}$$

Methanogenic archaeas digests, in the second stage, the VFA to produce methane and carbon dioxide, as described by the chemical reaction,

$$k_3S_2 \rightarrow x_2 + k_5CO_2 + k_6CH_4 \tag{2.19}$$

This model describes the methane production based on *acidogenic bacterias*, x_1, and *methanogenic bacterias*, x_2. Substrate is assumed to be described by *chemical oxygen demand* (COD), S_1, and *volatile fatty acids* (VFA), S_2.
Moreover, the model was developed for continuous reactors, so, it allows introduction of substrate along the time (S_1^{in} and S_2^{in}). Due to this introduction, mix is not perfect inside bioreactor is introduced a parameter of proportionality of experimental determination, α, where homogeneousness and other simplification are considered. Can also be found a parameter that relates the ratio of the influent flow rate and the reactor volume D (dilution rate of the influents).

This model has 5 equations to describe AD and the methane production.

$$\begin{cases} \dot{x}_1 = (\mu_1 - \alpha D)\, x_1 \\ \dot{x}_2 = (\mu_2 - \alpha D)\, x_2 \\ \dot{S}_1 = D\left(S_1^{in} - S_1\right) - k_1\mu_1 x_1 \\ \dot{S}_2 = D\left(S_2^{in} - S_2\right) + k_2\mu_1 x_1 - k_3\mu_2 x_2 \\ \dot{q_M} = k_6\mu_2 x_2 \end{cases} \tag{2.20}$$

Where:

- x_1 - is the concentration of acidogenic bacteria;

- x_2 - is the concentration of methanogenic bacteria;

- S_1 - is the concentration of chemical oxygen demand (COD);

- S_2 - is the concentration of volatile fatty acids (VFA);

- $\dot{q_M}$ - is the methane flow rate.

The variables μ_1 and μ_2 describe the micro-organism growth rate in an aqueous environment to the concentration of a limiting nutrient and are described by the Monod Law,

$$\mu_1 = \mu_{m1}\frac{S_1}{K_{S1}+S_1} \qquad \mu_2 = \mu_{m2}\frac{S_2}{K_{S2}+S_2} \tag{2.21}$$

Where parameters with index 1 refers to the acidogenesis and 2 to the acetogenesis processes.
Note that due to the low solubility of methane, model assume that all methane gets out of the bioreactor in a gas form with a mass rate that is proportional to the methanogenesis reaction.

2.4 Solving the Model

Since experimental data has many associated difficulties, such as noisy data and insufficient measurement frequency, among others, a noise-free simulation will be developed in first place to obtain a deeper understanding of the problem. Therefore, an initial study, using model (2.20) will be performed with the data presented in [5] to serve as the target results.
In order to generate this data a numerical solver of Ordinary Differential Equations (ODE) may be used. After some experiments, it was concluded that parameters values can produce situations of stiffness in the ODE model. For that reason the chosen solver able of managing such situation is important, so the search domain may be as wider as possible.
Another criteria for the choice of the solver has to do with the computational cost. Regardless of the optimization algorithm, the model will have to be solved oftentimes ended up being one of the most influential players in the computing time of the optimization process.
For the reasons presented, in SciPy (from Python) module can be found a solver that uses a Fortran 77 package [18] to solve both stiff and non-stiff equations. This package is called ODEPACK and, in

particular, SciPy makes use of the LSODA solver.
LSODA solve ODE systems with a dense or banded Jacobian when the problem is stiff, but it automatically selects between non-stiff (Adams) and stiff (BDF) methods. It uses the non-stiff method initially, and dynamically monitors data in order to decide which method to use. Thus, both prerequisites are met in a manner considered satisfactory [19].

To solve the ODE system, the initial conditions of the state variables must be known. In this case, the initial conditions used are [18],

- $x_1(0) = 0.2$ mg/L
- $x_2(0) = 0.8$ mg/L
- $S_1(0) = 74$ mg/L
- $S_2(0) = 93$ mmol/L
- $q_M(0) = 0.0$ mgl/L

The objective is to optimize the system parameters,

$$\vec{\theta} = \left[\mu_{m1}, K_{S1}, \mu_{m2}, K_{S2}, k_1, k_2, k_3, k_6\right]^T \tag{2.22}$$

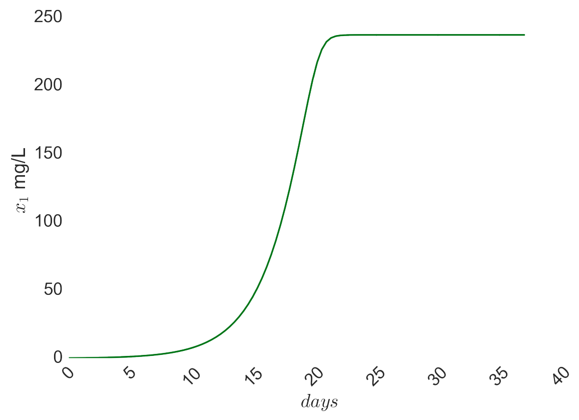
which for the target results have the following values,

- $\mu_{m1} = 4.2912 \times 10^{-1}$ day^{-1}
- $K_{S1} = 1.3065 \times 10^{1}$ mg/L
- $\mu_{m2} = 2.6493 \times 10^{0}$ day^{-1}
- $K_{S2} = 5.7127 \times 10^{2}$ mmol/L
- $k_1 = 3.1204 \times 10^{-1}$ mg COD/mg x_1
- $k_2 = 6.2776 \times 10^{-2}$ mmol VFA/mg x_1
- $k_3 = 3.1473 \times 10^{0}$ mmol VFA/mg x_2
- $k_6 = 2.7862 \times 10^{2}$ mL CH4/mg x_2

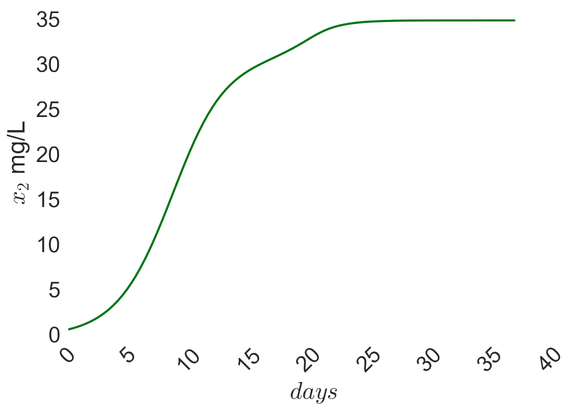
The model also has constants that are set to,

- $\alpha = 0$ day^{-1} - proportionality parameter of experimental determination $(0 \leq \alpha \leq 1)$;
- $D = 0$ day^{-1} - dilution rate of the influents;
- $S_1^{in} = 0$ mmol/L - influent concentrations;
- $S_2^{in} = 0$ mmol/L - influent concentrations.

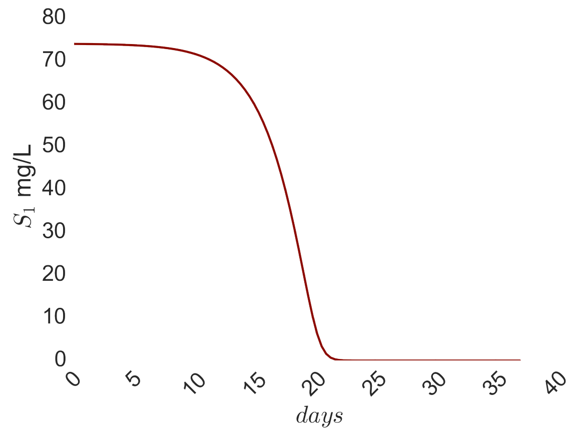
Although model (2.20) is valid for both batch, continuous and semi-continuous bioreactors, a batch reactor is considered here. The zero value of the previous constants occurs because there are no inlets in this type of reactors. Solving the model with the target parameters and presented initial conditions results in Figures 2.9 and 2.10.



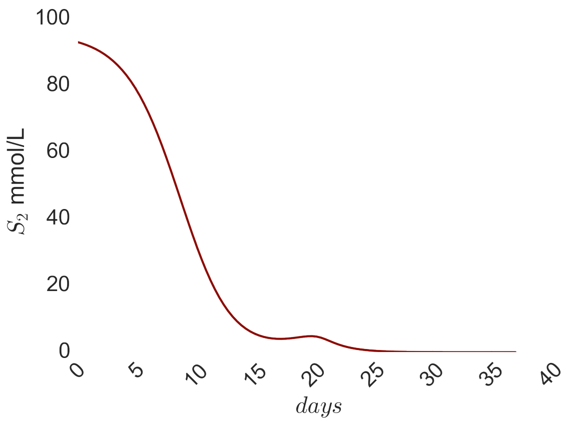
(a)



(b)



(c)



(d)

Figure 2.9: State variables of the target model.

As expected, since the model used is of batch type, acidogenic (x_1) and methanogenic (x_2) bacterias grow due to the bio-potential that exists in the substrate. On the other hand, Chemical Oxygen Demand (COD - S_1) and Volatile Fatty Acids (VFA - S_2) decrease with bacteria growth, going to zero after little more than 20 days. This coincides with the time that the bacterial population reach their maximum.

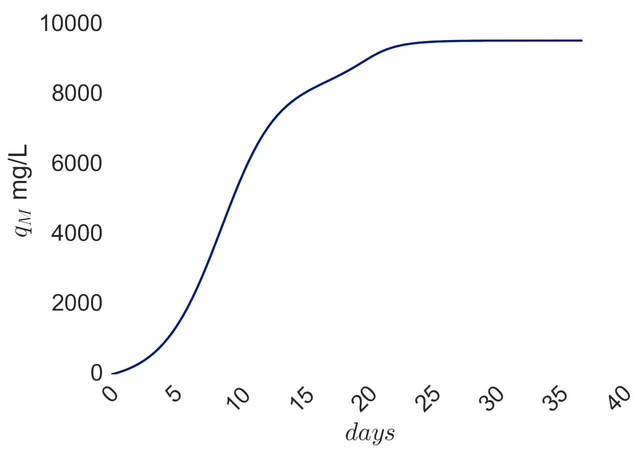


Figure 2.10: Output variables of the target model.

Methane is produced as a digestion by-product and is shown in figure 2.10. Since methanogenic bacterias are responsible for the methane production, the two evolutions are very similar in shape.

Chapter 3

Inverse Optimization

The development of a model should have into account [6]:

- **Simplicity**;
- **Causality**, i.e., it should be as representative of the most cause-effect as possible;
- **Identifiability**, i.e., all parameter should be possible to be evaluated;
- **Capability to make predictions**, i.e., it should be representative of what is being modeled.

Once a mathematical relationship (mathematical model) is achieved, its calibration is a necessary request, i.e., the parameters that are present in the model must be evaluated in order to be representative of a particular bioreactor. This is a complex task to be developed. For that reason, the use of prior knowledge of the bioreactor, experimental data, or any other knowledge source is necessary in order to perform an Inverse Optimization (IO) with some consistency in results.

Figure 3.1 shows the steps that should be performed in the modeling procedure.

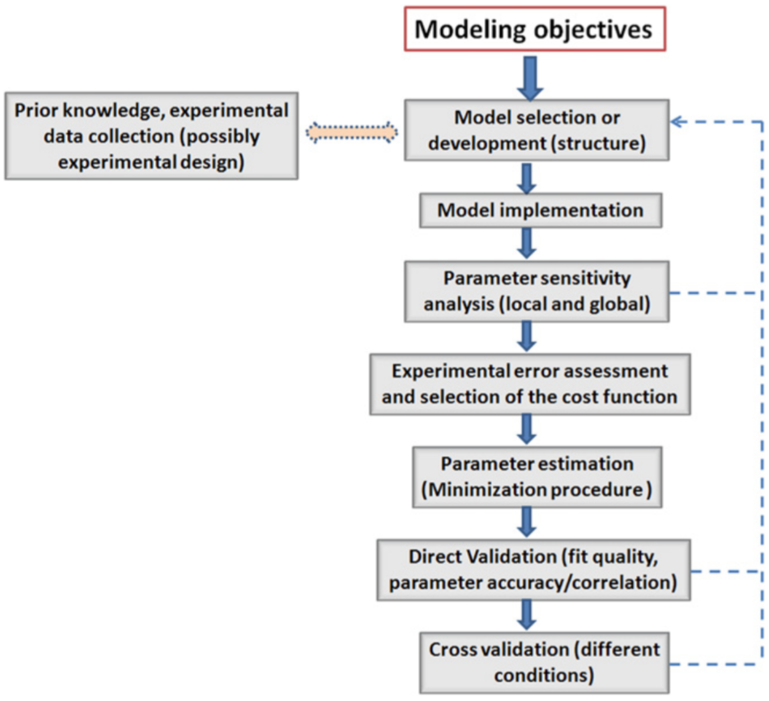

Modeling objectives
Prior knowledge, experimental data collection (possibly experimental design)
Model selection or development (structure)
Model implementation
Parameter sensitivity analysis (local and global)
Experimental error assessment and selection of the cost function
Parameter estimation (Minimization procedure)
Direct Validation (fit quality, parameter accuracy/correlation)
Cross validation (different conditions)

Figure 3.1: Steps for the bioreactor model calibration [6].

In this chapter some of the most important steps are exposed in order to collect capabilities to solve the AD problem.
In section 3.1 the mathematical formalization of optimization problems is presented as well as some of the difficulties that can be found in the optimization process and some of the most used cost functions. In this section the plot of the selected cost function for solving the Campestrini's et al. model is also presented in an attempt of getting sensibility to some of the difficulties that may be found in further analyses. In the next section, 3.2, the Genetic Algorithms (GA) is presented. This is followed by the presentation of some of the difficulties inherent to the method. In order to overcome some of those, an improvement known as hybridization is also introduced (section 3.3).
At last, in section 3.4 some benchmark functions are used to test, validate and understand what to expect from GA and how are they suitable for each kind of problem. This section ends with the code validation using Campestrini's et al. model where k_6 parameter is estimated. Among these analyses a comparison between Basic Genetic Algorithms (BGA) and the new hybridization that was developed in this work, the Neighbored Genetic Algorithm (NGA), are compared.

3.1 Optimization Problem

Generally, the development of a physical model follows three major steps [20]:

1. **System parametrization**: Where a set of parameter which describes the system are established;
2. **Modelation**: At this stage, the mathematical relationship that predicts the system response is developed, among with the parameters obtained before;
3. **Inverse Modelation**: After a mathematical model is established, it is, in principle, possible to extrapolate it for other similar systems. It is, however, necessary to identify the new parameter values that will describe that system. To achieve this, is necessary to develop an inverse optimization problem (IOP).

In a general sense, when establishing an optimization problem is good practice to be aware of some difficulties and steps that helps to get consistency in the results.
Among some of them are:

- Problem dimension (also known as the *curse of dimensionality*)
- Parameters identifiability and sensibility
- Choice of optimization algorithm
- Choice of objective function

Furthermore, Optimization Problems (OP) may be categorized according to Figure 3.2.

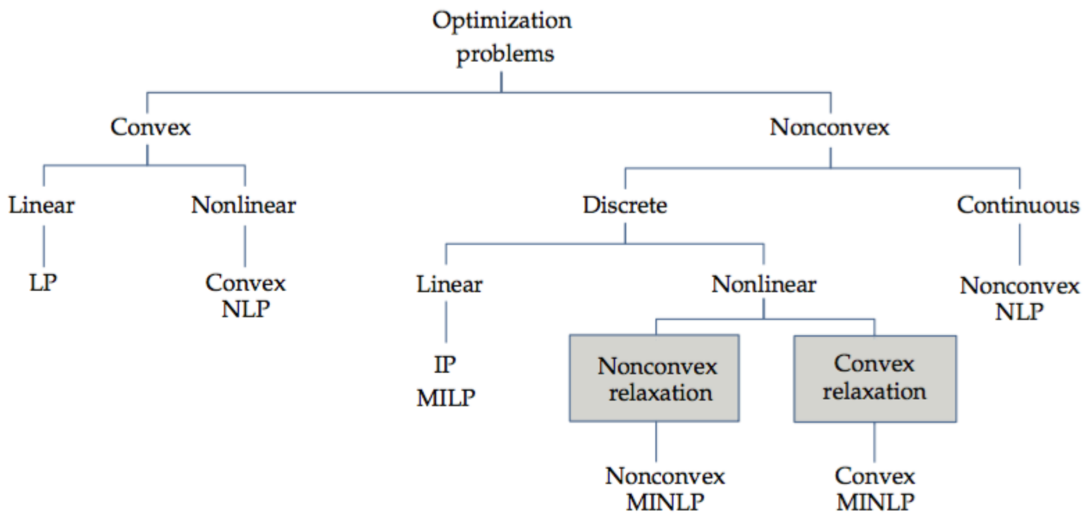


Figure 3.2: Optimization problem types overview [21].

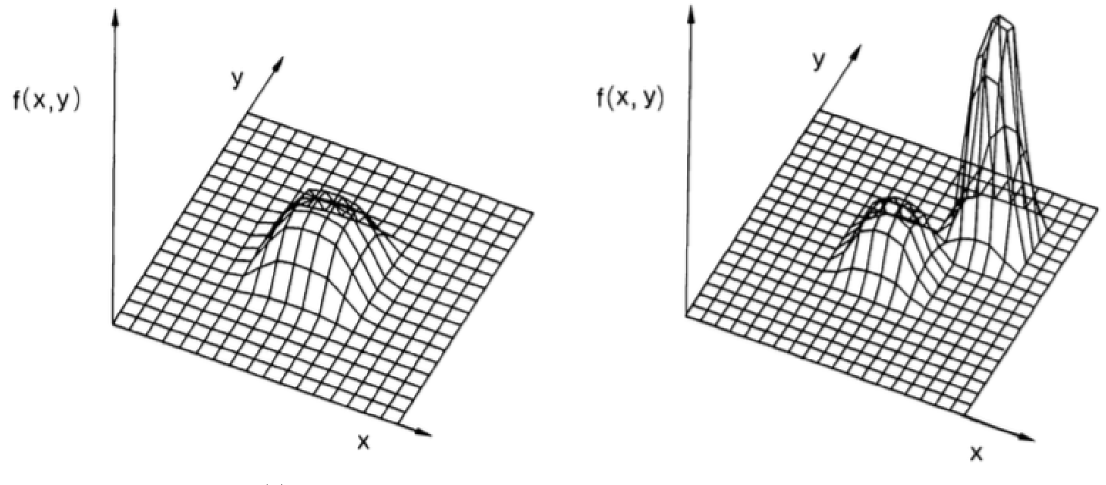
In the chart NLP stands for nonlinear programming problems and MINLP for mixed-integer nonlinear programming problems. The parameter problem of identification in AD models is a nonconvex NLP, as it will be shown.

The choice of the optimization method is of extreme importance to achieve meaningful results in the shortest computational time. To solve the IOP one of following methods may be chosen:

- **Deterministic**, or exact, approaches, make use of analytical properties of the problem, so in each step solution converges to an optimal. Typically, in this methods at least one initial guess must be given to start the algorithm. Moreover, in equal conditions (such as tolerance, number of iterations, etc), the algorithm always returns the same solution to each initial guesses;
- In **stochastic** algorithms, such as evolutionary algorithms (EA), randomness is used to search for the optimum solution. In this case due to intrinsic randomness of the algorithm, for the same initial guess different solutions may be found.

Despite a deterministic method always find an optimal solution (even if not global), unlike heuristic methods that only search for a good one, exact methods may be computationally expensive in difficult problems and, in some cases, inapplicable [22]. In particular, IOP are some of the most challenging to solve.

Figure 3.3 presents a very simple example that shows one of the difficulties of deterministic methods.



(a) (b)

Figure 3.3: Examples to distinguish between convex (figure 3.3a) and non-convex (figure 3.3a) functions.

On the left figure a convex hull is represented (figure 3.3a) and, on the right, a non convex hull (

(figure 3.3b)). In 3.3b, the initial guess may be such that search will find a local maximum instead of the global one. Which initial guess should be the chosen, is a deterministic problem that may only be answered by trial and error. This example easily demonstrates that for more complex problems this may be almost impossible to achieve.
It is clear that, for each of those searches, initial guesses are of major importance to achieve the objective and, in more complex problems, they can be not trivial to find. Moreover, in IOP it may be of interest to find one global maximum/minimum value, multiple or all.
Figure 3.4 illustrates how the type of problem influences the need for robustness of the algorithm and its impact on efficiency.

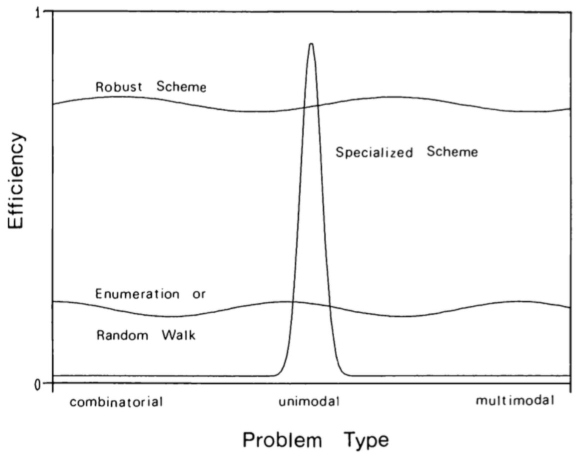


Figure 3.4: Influence of the problem type in the algorithm exigency and efficiency [23].

For multimodal problems the target may be to obtain multiple or all minima of a cost function (e.g., eigenvalues determination). In particular, if only there is one global minimum, than the problem is called unimodal. Notice that for those cases, robustness is a prerequisite, nevertheless it brings efficiency with it. If, solution precision is not very important and what is intended is a close enough idea of the solution, than a specialized scheme may be used. This way robustness may be relaxed and a slight knowledge of the problem is achieved anyway.
In combinatorial problems the solution can be one of a multiple finite possibilities (e.g., the classical traveling salesman problem).

Some of the deterministic methods that may be used are [24]:

- Gradient descent;
- Conjugated gradients

- Branch-and-bound;
- Outer-approximation;
- Cutting planes;
- Decomposition.

On the other hand, examples of stochastic methods are [24]:

- Random search (ex: *Monte Carlo*);
- Genetic algorithms;
- Particle swarm algorithms;
- Clustering algorithms;
- Multi-level single linkage methods.

Independently of the algorithm used, the good establishing of the mathematical optimization problem is important. In particular, for the Campestrini et al. case:

$\vec{\theta}$ - the vector of *variables*, also called *unknowns* or *parameters*;

$J(\vec{\theta})$ - the *cost function* (also called *cost function* or *loss function*), a function of $\vec{\theta}$ that, in general, is pretended to minimize;

t - the time period of bioreactor collected data;

$y^{\text{target}}(t)$ - is the target data to be optimized;

$y(\vec{\theta}, t)$ - is the solution for the $\vec{\theta}$ guess;

$$\min \quad J(y^{\text{target}}(t), y(\vec{\theta}, t)) \quad \text{subjected to} \begin{cases} \vec{\theta} \geq 0 \\ y(\vec{\theta}, t) \geq 0 \end{cases} \tag{3.1}$$

The choice of the cost function is another important step because sensitivity can be greatly improved for the parameters estimation with its choice. Some of the considerations that should be taken are:

- Whenever possible, it should produce a convex hull;
- For a search space, it should not be too steep nor too flat.

According to [6], some of the most used cost functions, for the optimization problem solved in this field, are:

$$J(y(\vec{\theta},t),y^{\text{target}}(t)) = \min \sum_{i=1}^{N} \left(y(\vec{\theta},t_i) - y^{\text{target}}(t_i) \right)^2 \tag{3.2}$$

$$J(y(\vec{\theta},t),y^{\text{target}}(t)) = \min \sum_{i=1}^{N} w_i \left(y(\vec{\theta},t_i) - y^{\text{target}}(t_i) \right)^2 \tag{3.3}$$

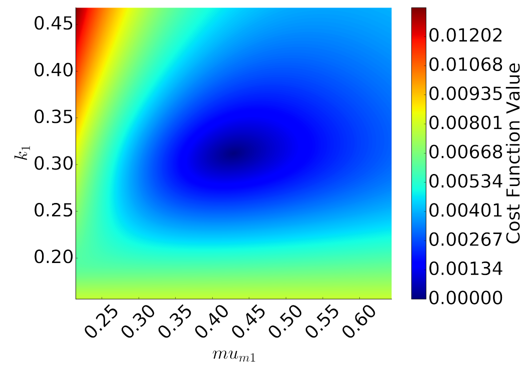
$$J(y(\vec{\theta},t),y^{\text{target}}(t)) = \min \sum_{i=1}^{N} \left(\frac{y(\vec{\theta},t_i) - y^{\text{target}}(t_i)}{y^{target}(t_i)} \right)^2 \tag{3.4}$$

$$J(y(\vec{\theta},t),y^{\text{target}}(t)) = \min \sum_{i=1}^{N} w_i \left(\ln y(\vec{\theta},t_i) - \ln y^{\text{target}}(t_i) \right)^2 \tag{3.5}$$

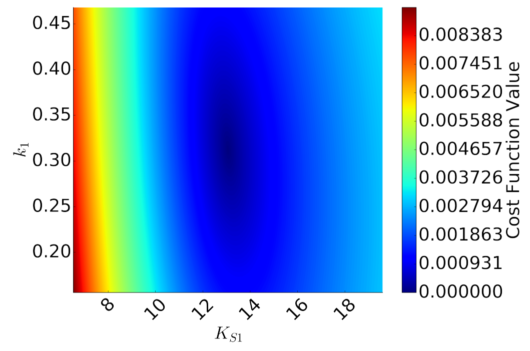
So, for the purposes of this work, the cost function:

$$J(\bar{y}(\vec{\theta},t),y^{\text{target}}(t)) = \frac{1}{N-1} \sqrt{\sum_{i=2}^{N} \left(\frac{y^{\text{target}}(t_i) - \bar{y}(\vec{\theta},t_i)}{y^{\text{target}}(t_i)} \right)^2} \tag{3.6}$$

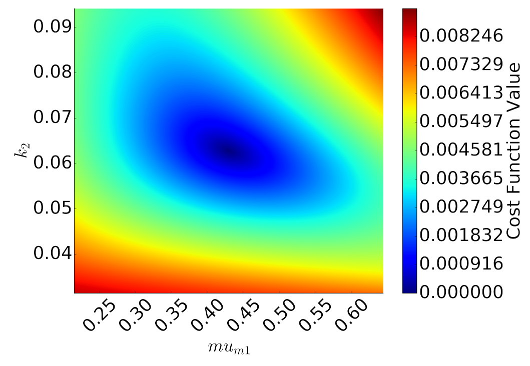
was considered, which is similar to (3.4).
Since, methane is the easiest variable to be measured, it will be the one used as output variable for the evaluation of the cost function. Since the initial condition for this variable is zero, it is skipped and the sum is started at the second value.
In order to have further knowledge of the optimization architecture developed, cost function is plotted as function of pairs of parameters (figures 3.5 to 3.8). This figures are a tentative of understand how parameters variation influence convexity of the cost function.
These plots will be separated in three group - the ones that are expectantly easy to find, those that have a clear minimum but that may be harder to identify and a group of parameters that can be extremely hard to determine and/or can be impossible if data has noise.



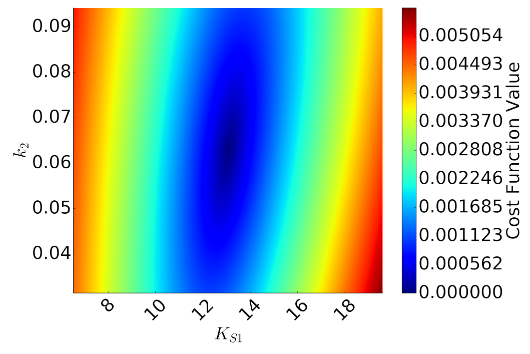
(a)



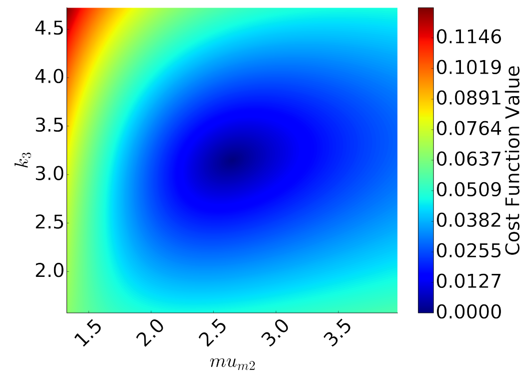
(b)



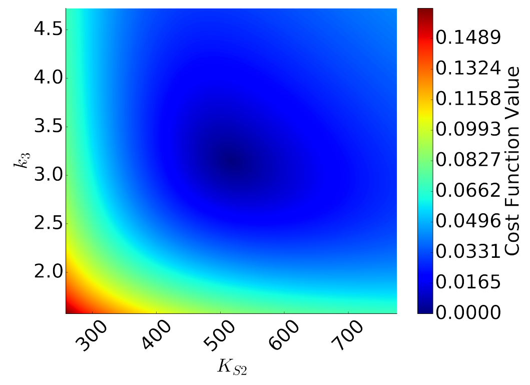
(c)



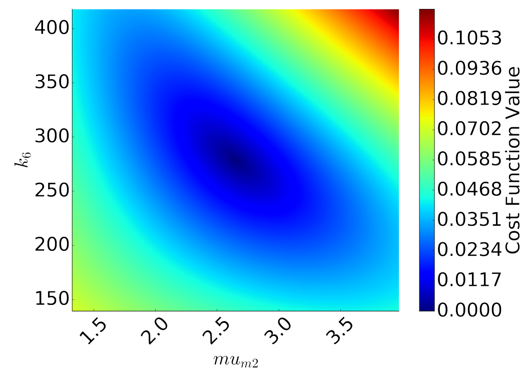
(d)



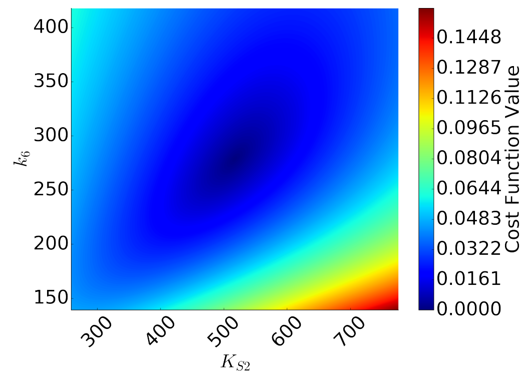
(e)



(f)



(g)



(h)

Figure 3.5: Group of cost function value varying pairs of parameters which are considered of easy identification.

Figure 3.5 shows multiple combinations of parameters that produces a convex hull with a clear minimum. This means that, for example according to figure 3.5a, if the k_1 value is found, than μ_{m1} is automatically identified as well, moreover the other way around is also true (for the presented domain at least).

These figures show other relevant information, that is, after the parameter field variation, the greater value of the cost function is small. This may prove to be a difficulty, since despite the cost function have a low value, the parameters may be far from the goal.

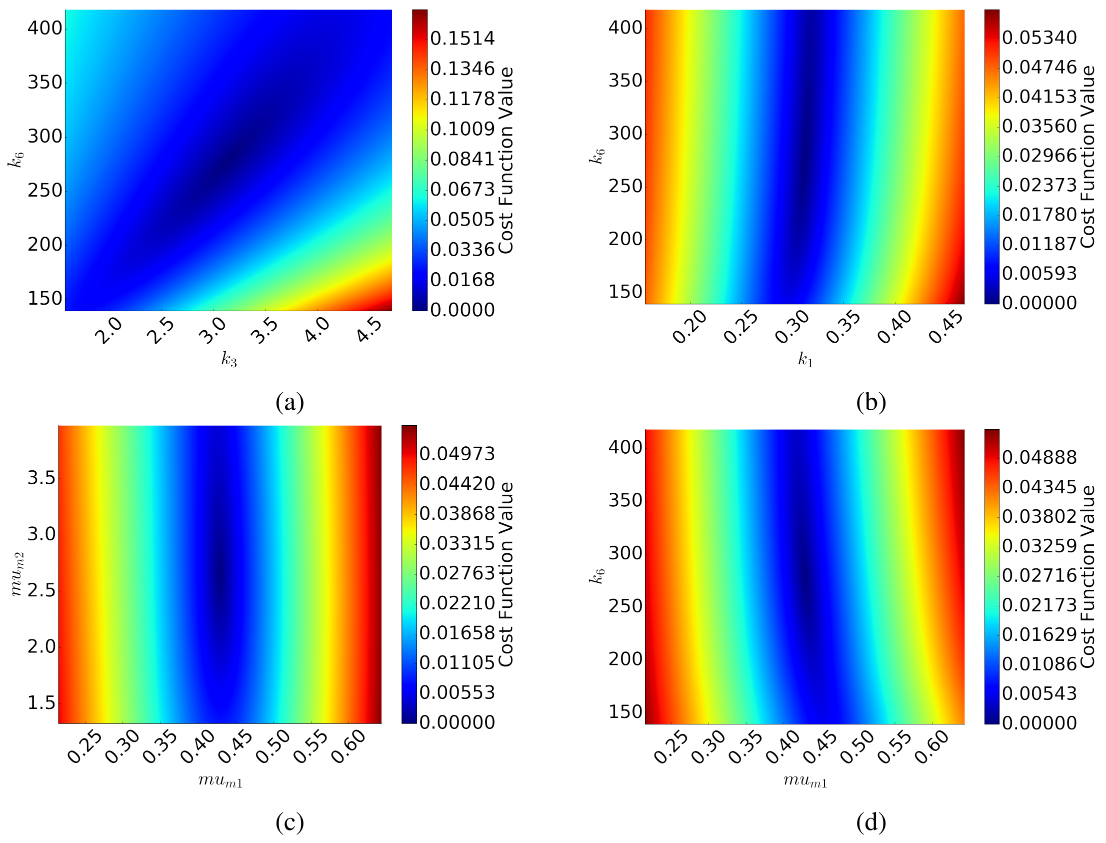


Figure 3.6: Group of cost function value varying pairs of parameters which are considered not so easy of identification.

Figure 3.6 shows cases where convexity starts to be blurred. This proves to be a difficulty when optimizing because cost function has low values for multiple combinations of parameters value. Despite this difficulties, the minimum is still visible in the representation so its identification may be hard but still assumed possible to find.

On the other hand, in the following figures, minimum is much more unclear and are placed or in a steep valley or in a flat area. This is the extreme case of the the one presented before, i.e., minimum is unclear and is blurred along a wide range of parameters combination.

In the case of figure 3.8a, for example, the k_3 may be found, since that axis shows a clear minimum, but the search of K_{S1} may be hard of even impossible, since there are an infinite situations that produces an extremely low cost function value.

For experimental data this results may look even more extreme with presence of noise.

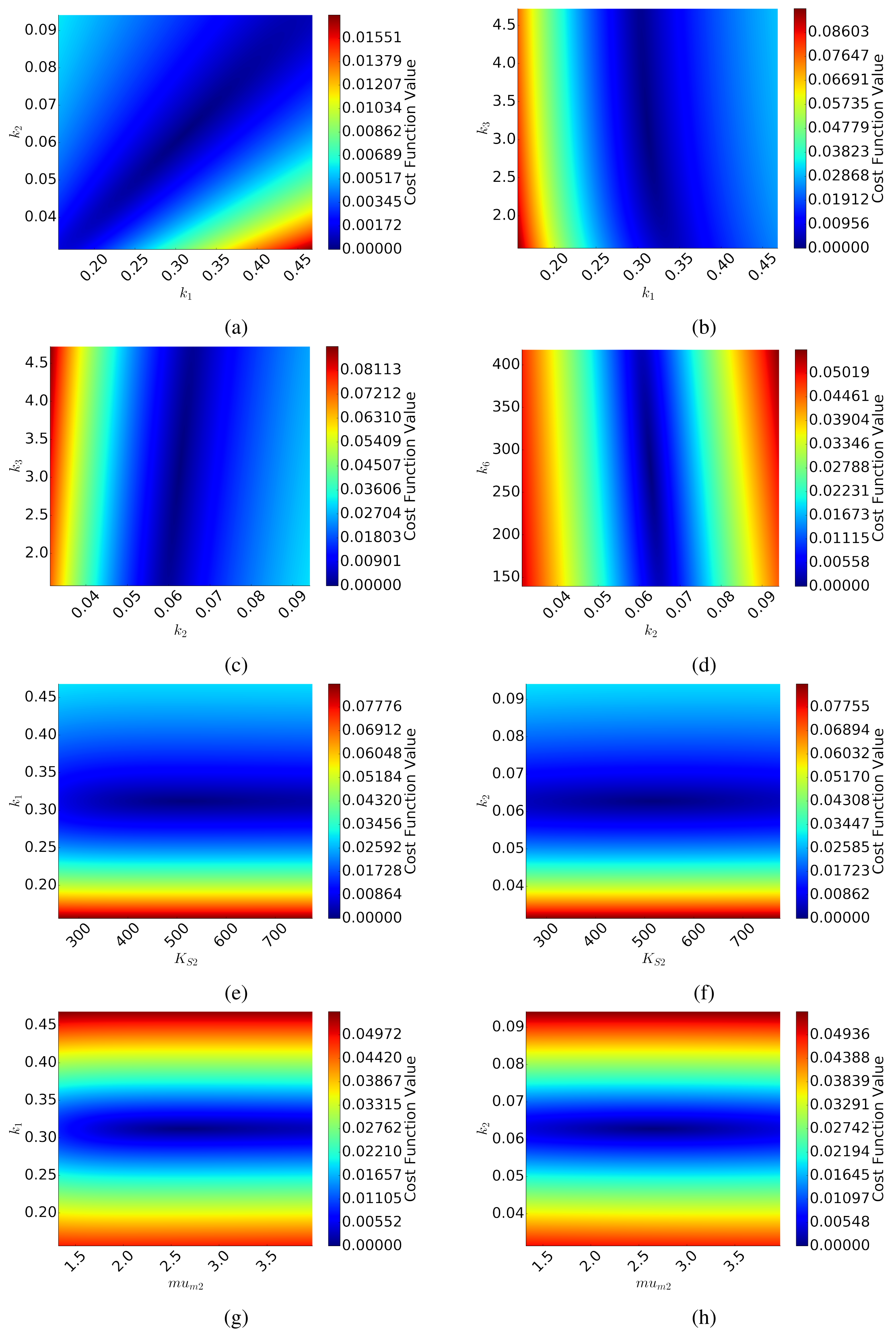
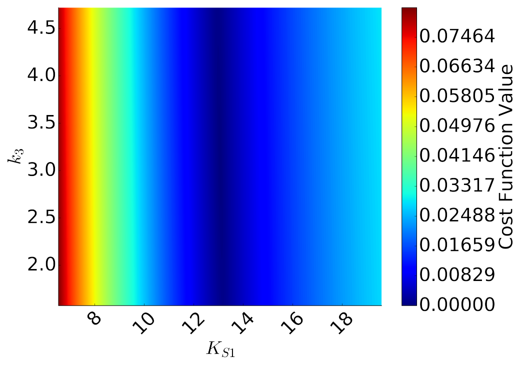
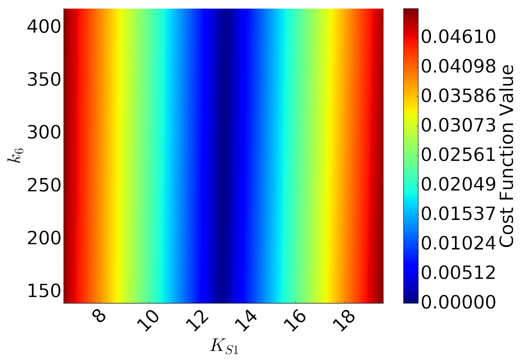


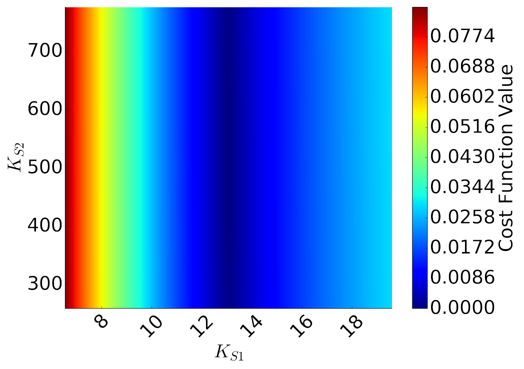
Figure 3.7: Group of cost function value varying pairs of parameters which are considered hard to find.



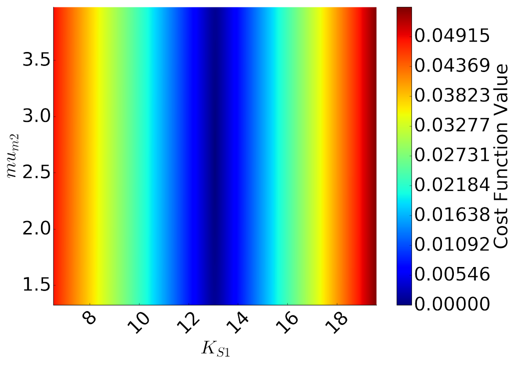
(a)



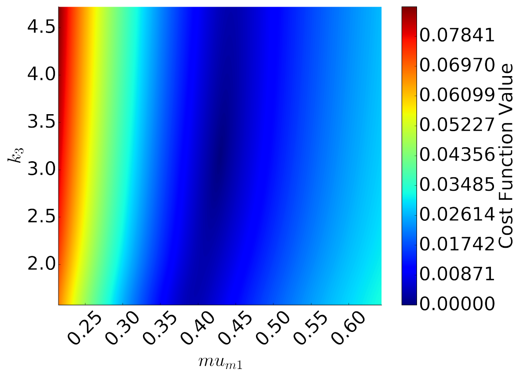
(b)



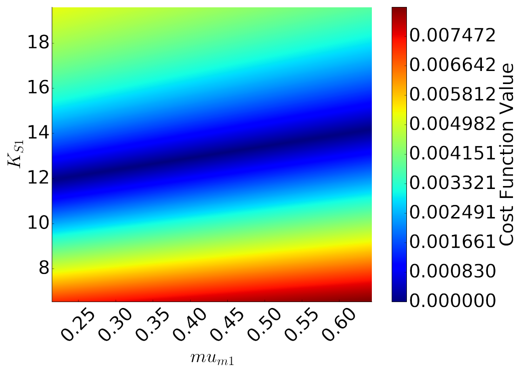
(c)



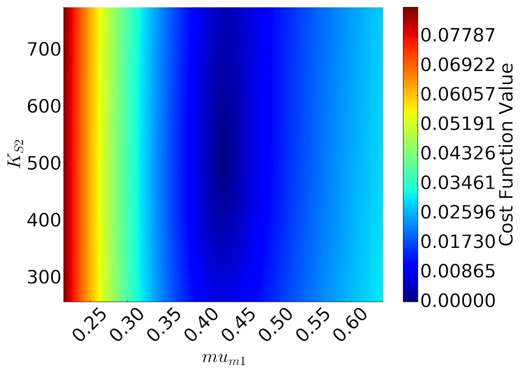
(d)



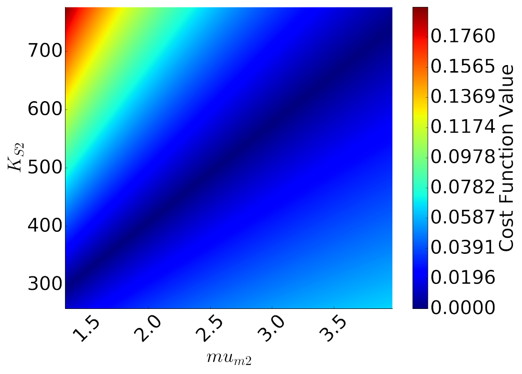
(e)



(f)



(g)



(h)

Figure 3.8: Group of cost function value varying pairs of parameters which are considered hard to find (continue).

The previous figures suggests that for the range considered there may exist a single local minimum

for, at least, most of the cases, but this conclusion can not be exhaustive. For that reason further study of the model must be performed.
If the existence one minimum is unique for the observable variables, than the model is said global identifiable. On the other hand, if there exists multiple global minimums, than, the model, is said locally identifiable. Finally in the case that there exists an infinite number of solutions that produces global minimums than the model is said non-identifiable.

In the case of the previous figures, it is shown that there may not exist one single combination of parameter values for which the minimum is unique, so is expected that the identification of all parameters when only q_M is observable may be not achievable, nevertheless, to confirm this, an identifiability analysis is performed.
One, using GENSSI, a MatLab based software that produces this analyses using Lie's Algebra. And, the other one, using DAISY (Differential Algebra for Identifiability of Systems) which is an identifiability software assistant that uses derivatives to show if there is identifiability and, in that case if it is local or global.

The difficulties associated with this approach is that the order of Lies derivatives must have the same number of parameters in analysis and the greater the order the greater the computational cost is. In this case, since there are 8 parameters for identification then it is needed days or even weeks of computation which is impossible to perform in the time window of this work. For that reason, the approach that will be taken is to take steps in that direction.

From the first identifiability analysis, using GENSSI, is sought to know if all parameters are identifiable when all state variables are observable. Results shows that, it is possible and that there is a single global minimum. This means that a global optimization process, with more or less effort, should reach, at some point, this global minimum.
As a second analysis, if S_1, S_2 and q_M are the only ones observable, than, once again all the 8 parameters are globally identifiable.
In the case of q_M being the only observable quantity and assuming that μ_{m1}, K_{S1}, μ_{m2} and K_{S2} are fixed, than parameter identifiability analysis shows that, unlike k_6 which is globally identifiable, all remaining parameters are only locally identifiable.
When running GENSSI with q_M variable observable and all parameters free, software has been unable to retrieve an answer in time, even after a few days trying to find an answer. For that reason the

multiple runs that will be used for the comparison of BGA and NGA performances will be used, as well, to estimate about identifiability using a statistical approach.

3.2 The Role of Genetic Algorithms

Among the available heuristic methods are the Genetic Algorithms (GA). They are a particular algorithm from within the Evolutionary Algorithms (EA) that has as analogy of the species evolution with the survival of the best fitted individuals.

Some reasons to choose EA over deterministic algorithms are [25]:

- Global minimum search method;
- No need neither of Jacobian nor of Hessians knowledge;
- Easily implemented;
- Robust response when changing circumstance;
- Depending of the problem approximation produced by EA may be sufficient.

All EA shares the same principles, i.e., all simulate an evolution of a population making use of operators such as:

- Selection;
- Mutation;
- Reproduction.

In Figure 3.9 a flow chart where general EA is shown.

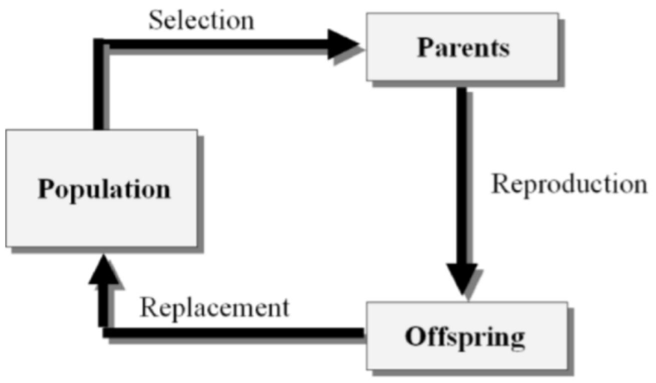


Figure 3.9: Flowchart of an Evolutionary Algorithm [25].

In particular, GA are based in the theory evolution of species in which the survival of the strongest is mathematically formulated as the best solution contribute for the general population evolution.
Each individual of the population (called *structure* in artificial GA) has a chromosome, $\vec{\theta_i}$, which is a sequence of *genes* (in natural systems this sequence is called genotype). The value that each gene takes is called *allele*. A position in the chromosome, is called *locus*, and is associated to each gene.
Any GA starts with an initial population (which, in the context of this work, are multiple hypothesis of the parameters estimation) and after this a sequence of operations is processed in order to evolute to an optimal solution.
In order to keep generality in GA code implementation, ease the GA operations since they are random base and are numerically more manageable and to have more suitable numerical representation, for the purposes of this work, all parameters will be constrained within a domain of 2^{-10} and 2^{10}, so it will be considered for normalized values, θ_i^{norm} between $[0, 1]$, the expression,

$$\theta_i = 2^{\theta_i^{\mathrm{norm}}(\mathrm{Dom}_{\mathrm{max}} - \mathrm{Dom}_{\mathrm{min}}) + \mathrm{Dom}_{\mathrm{min}}} \tag{3.7}$$

Where $\mathrm{Dom}_{\mathrm{min}}$ and $\mathrm{Dom}_{\mathrm{max}}$ refers a maximum and a minimum of the search domain of the parameter i.

In order to get a large range of search, the domain will be kept between,

$$\mathrm{Dom} = [-10, 10]$$

Due to this large range of parameters ODE can becomes stiff in some cases, so the choice of a solver able to handling this issues is a requirement.

Figure 3.10 gives an overview of the major operation of GA with a flow chart diagram.

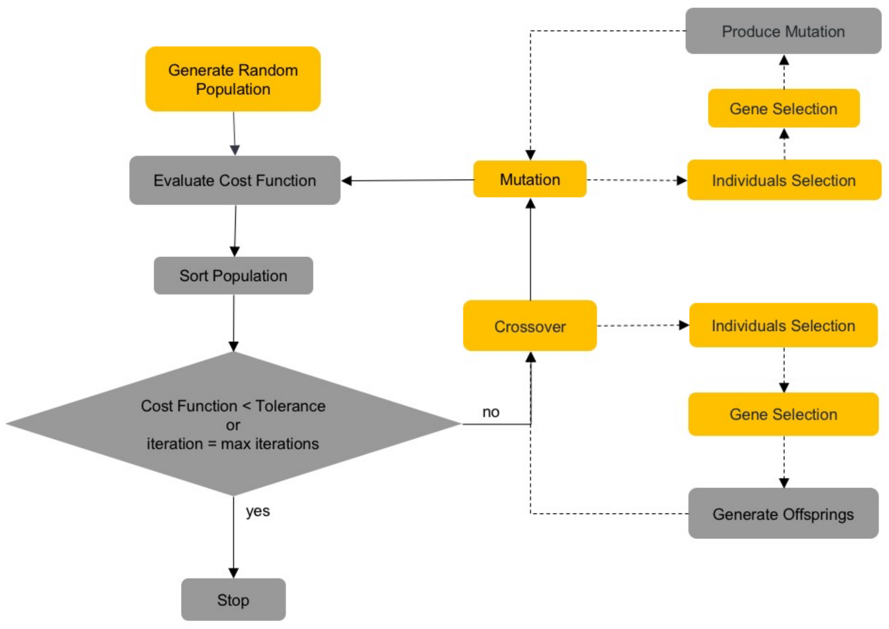


Figure 3.10: Flowchart of Genetic Algorithms.

The major differences that can be found in the literature are applied to operations presented in yellow balloons - population generation, selection, crossover and mutation. In subsequent subsections this operations are detailed.

3.2.1 Population Generation

Population may be generated multiple ways, it can be imposed based on prior knowledge, randomly or even a mix between the two.

In this work an initial population, Θ, of n individuals is randomly generated where each individual, $\vec{\theta}$, represents a potential solution and is composed of m allels which code the parameters that need to be estimated. In the case of the modeling of a bioreactor each allel represents a parameter of that model. Each of the n individuals is ranked using the cost function $(J(\vec{\theta}))$ and then sorted according their fitness.

3.2.2 Selection

Once the population is sorted, individuals must be eligible for the cross operation. For that, a selection criteria must be established.

Figure 3.11 shows a diagram of Abuiziah et al, [26], review on GA strategies for selection operations.

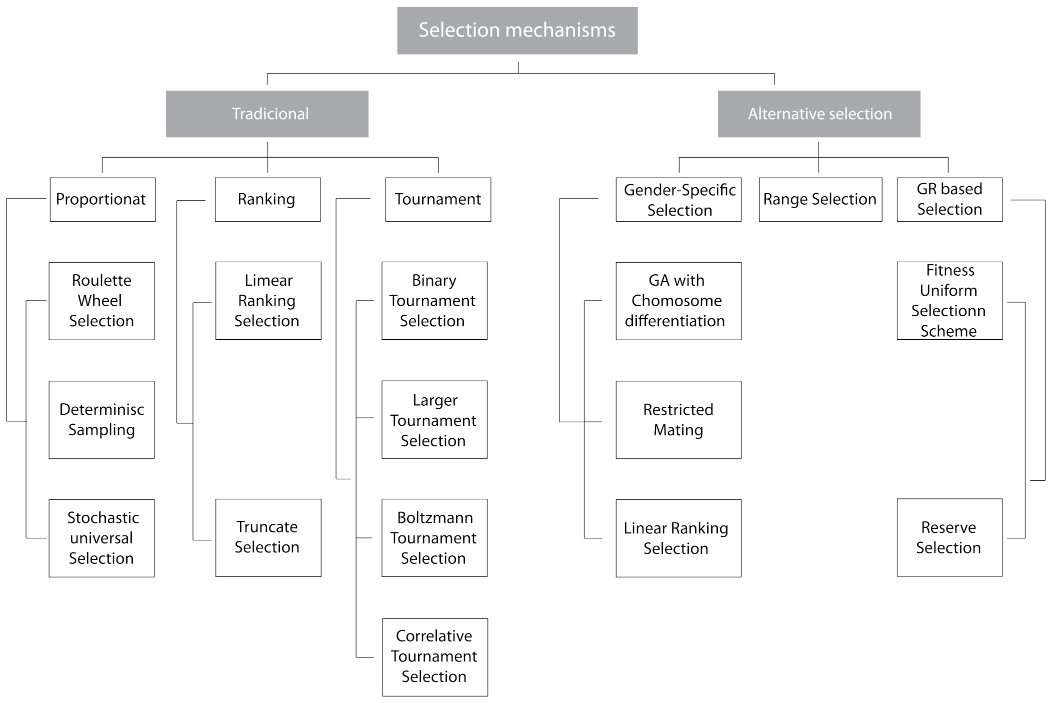


Figure 3.11: Review on selection approaches (adapted from [26]).

One of the possible ways of doing that, is using the Roulette Wheel criteria, where it starts with the weighting of the rank position. This ranking is generated by the probability of a subject being selected and is evaluated by,

$$P_i = \frac{n-i+1}{\sum_{j=1}^{n} j} \tag{3.8}$$

The main idea of equation (3.8) is to have greater probability (P) in the best individuals so they will be selected more often than the worsts for crossing.

Based on this, to proceed with selection, a vector of dimension n is generated with random numbers within the $[0,1]$. Each individual is then compared with probability of equation (3.8) and if that random value is immediately greater than the ranked value of that individual it is stored in $\vec{s}$ for the crossing procedure. For each random number r_i equation (3.9) shows the way to construct the selection vector $\vec{s}$.

$$s_i = \begin{cases} 1, & P_1 \leq r_i \\ j, & P_{j-1} \leq r_i \leq P_j \end{cases} \tag{3.9}$$

3.2.3 Cross-over

Crossing is another operation that is one of the primal operations of GA when generating a new improved population. It is responsible for combining (mating) two individuals to produce new ones (offspring). The way it is performed may vary from author to author, in Abuiziah et al, [26] and Jorge Magalhaes-Mendes [27] some of them can be found:

- Single point crossover;
- Two points crossover;
- Intermediate (uniform) crossover;
- Arithmetic crossover;
- Heuristic crossover;
- Ring crossover.

In this work the single point strategy will be used, so, in pairs, those selected individuals, $\vec{s}$, may produce two childs and then perish. For that to happen, a random number is generated and compared to a user established crossing probability (usually it takes values around 75% and in this work it will be considered 80%). If the comparison of the random number is greater than the crossing probability than individuals are kept in population without any change, but if it is not, than the crossing procedure is initiated.

For that, considering any two parents i and j and an a allel, procedure is performed as follow:

1. Generate an integer random number, l, within the allel length;
2. Generate a new random number for the crossing weighting;
3. The allels in both childs in l's position will be replaced by:

$$\begin{aligned} a_i^{\text{child}} &= a_i^{\text{parent}}\beta + a_j^{\text{parent}}(1-\beta) \\ a_j^{\text{child}} &= a_j^{\text{parent}}\beta + a_i^{\text{parent}}(1-\beta) \end{aligned} \tag{3.10}$$

4. From the the allel $a + 1$ forward allels of parent i are swapped with the j's ones.

Figure 3.12 shows an illustrative example of crossing with a population of 8 allels.

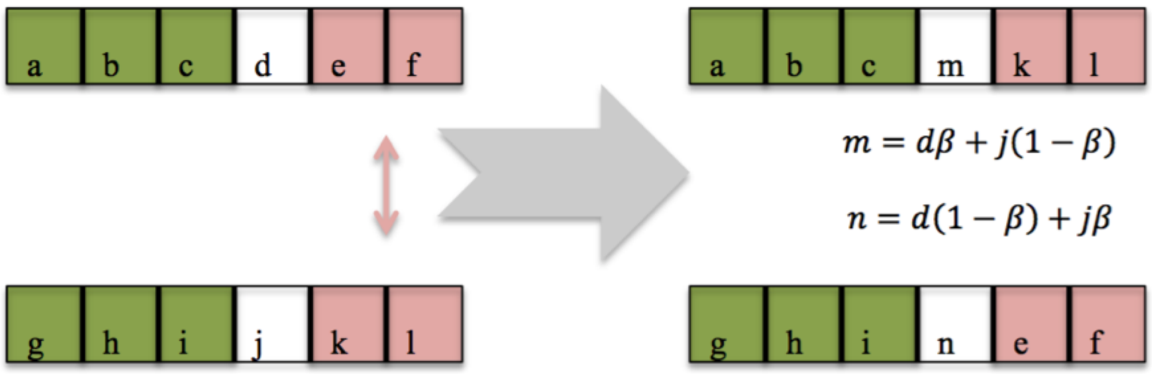


Figure 3.12: Crossing procedure.

In the example presented in figure 3.12 the random integer is 4 so all genes before are kept, the fourth is weighted between both parents with a coefficient β, which is a random number within [0, 1], and the remaining genes are swapped. In the end, each pair has generated another pair and a new population was obtained.

3.2.4 Mutation and Elitism

One way to avoid population's saturation is making use of mutation [27]. In this process is established a probability to evaluate if the mutation is performed in each subject (typically around 30%).
Case a generated random number is lower than the stipulated the procedure starts by randomly choosing a gene and after that is substituted by a random new one using a standard deviation of 0.2 around the old one.
In order keep the best estimate so far, elitism is applied. It keeps the fittest individual from iteration to iteration. Moreover, because population size must be kept, this individual replaces the worst one from the new iteration.

3.3 Hybridization of EA

Although EA (and GA, in particular) it is a very popular optimization method, it is not well suited for fine tuning search [28]. Its extensive use has led to the development of improved GA that try to overcome the practical limitations of the traditional GA implementation proposed by Holland [29]. Examples of such attempts are CHC GAs [30], adaptive GAs [31], niche GAs [28], hybrid GAs

[32] and MO-GAs [33]. Hybrid GAs in particular, and, memetic algorithms in general, combine evolutionary algorithms with local refinement. For certain type of problems, memetic algorithms are both more efficient and more effective than traditional evolutionary algorithms [34]. In the specific case of GAs, one can integrate a traditional GA implementation with local search heuristics. This integration can introduce new genes, therefore diversity, and help avoid genetic drift [35]. Many are the examples of hybridization of GAs with different local search algorithms ([36], [32], [37]). Due to the difficulties indicated, hybridizations of EA have been widely used trying to improve the rate of convergence and to get a closer approximation to solution [38].

As reported in the literature, different methods of performing hybridization of EA can be found. Some strategies of combining EA with other methods involve [25]:

- Hybridization between an evolutionary algorithm and another evolutionary algorithm (example: a genetic programming technique is used to improve the performance of a genetic algorithm);
- Evolutionary algorithms assisted by neural network;
- Evolutionary algorithm assisted by fuzzy logic;
- Evolutionary algorithm assisted by particle swarm optimization (PSO);
- Evolutionary algorithm assisted by ant colony optimization (ACO);
- Evolutionary algorithm assisted by bacterial foraging optimization;
- Hybridization between evolutionary algorithm and other heuristics (such as local search, tabu search, simulated annealing, hill climbing, dynamic programming, greedy random adaptive search procedure, etc)

Moreover, the way these strategies are implemented with the EA can be found in different ways. Figure 3.13 shows the fundamental architectures to join the hybridization technique.

In case of figure 3.13.a an independent procedure if performed with the information that is outputted from the EA (for example, a local search to increase accuracy may be performed using the estimation obtained from EA). Figure 3.13.b shows a similar approach but, in this case, EA is performed after some intelligent paradigm (for example, EA initial population may follow some rule obtained from any source of knowledge about the cost function). A cooperative architecture can be found in figure 3.13.c, where some kind of information may be shared between the EA and the intelligent paradigm

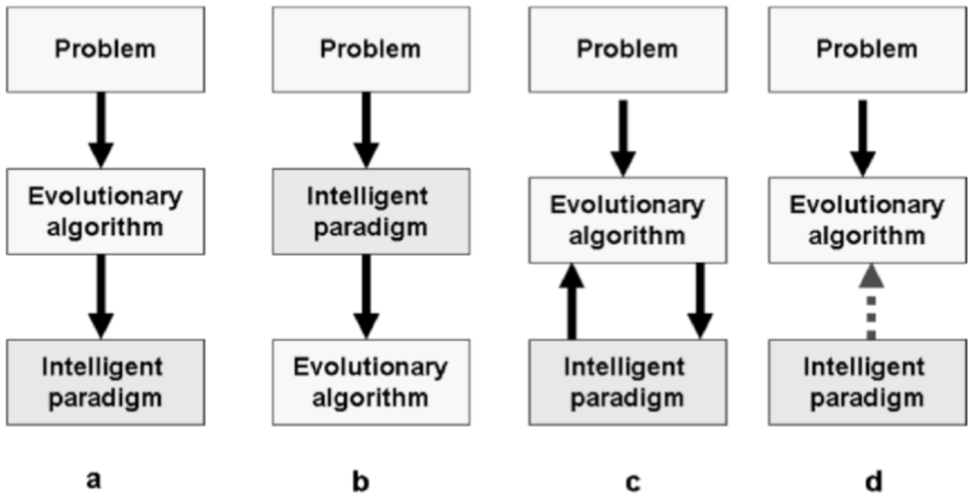


Figure 3.13: Fundamental architectural strategies to hybridize GA [25].

in order to improve the search process. On the other hand, the other strategy may be blind to what is occurring in the EA but still assists (figure 3.13.d).

Some of the intelligent paradigms found in literature are [25]:

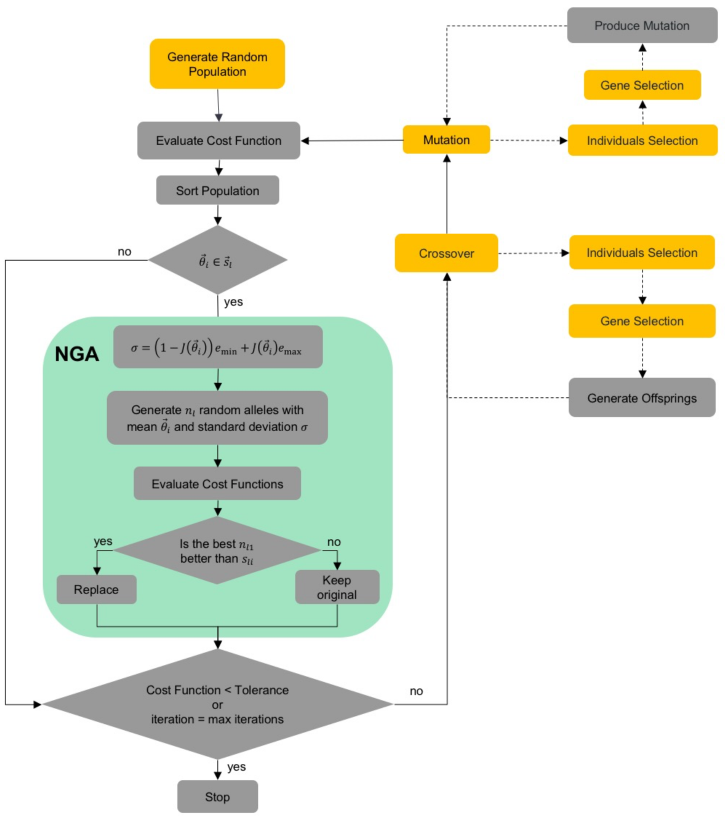
- The solutions of the initial population of EA may be created by problem-specific heuristics;
- Some or all the solutions obtained by the EA may be improved by local search. This kind of algorithms are known as memetic algorithms;
- Solutions may be represented in an indirect way and a decoding algorithm maps any genotype to a corresponding phenotypic solution. In this mapping, the decoder can exploit problem-specific characteristics and apply heuristics etc;
- Variation operators may exploit problem knowledge. For example, in recombination more promising properties of one parent solution may be inherited with higher probabilities than the corresponding properties of the other parent(s). Also mutation may be biased to include in solutions promising properties with higher probabilities than others.

In particular, Memetic Algorithms (MA) are a population based algorithm that uses separated individuals for learning or local improvement procedures on problem search. In the literature, MA are also referred as Baldwinian evolutionary algorithms (EA), Lamarckian EAs, cultural algorithms, or genetic local search [25].

Multiple strategies of MA implementation may be found, usually direct local searches are used [25]. The problem with this strategy is that, since it is based on a deterministic method, the convergence for the same minimum can be frequently performed with no improvement of the overall population.

Another frequent strategy that appears in literature is the use of the information of individuals, such as the cost function value, to redefine the GA operation probabilities (roulette wheel, cross probability, mutation probability, etc) [25].

In this work, a new approach to the MA is developed and presented. It has the form of a controlled scattered local search. What is expected with this is that, the GA may be able to avoid saturation of the population and improve convergence (figure 3.14).



Generate Random Population
Evaluate Cost Function
Sort Population
$\vec{\theta}_i \in \vec{s}_l$
no
yes
NGA
$\sigma = \left(1 - J(\vec{\theta}_i)\right) e_{\min} + J(\vec{\theta}_i) e_{\max}$
Generate n_l random alleles with mean $\vec{\theta}_i$ and standard deviation σ
Evaluate Cost Functions
Is the best n_{l1} better than s_{li}
yes
no
Replace
Keep original
Cost Function < Tolerance or iteration = max iterations
no
yes
Stop
Produce Mutation
Gene Selection
Mutation
Individuals Selection
Crossover
Individuals Selection
Gene Selection
Generate Offsprings

Figure 3.14: Flux flow diagram of the NGA algorithm.

This approach will be called Neighbored Genetic Algorithm (NGA) and the main idea of the method is to generate a number n_l possibilities around the current solution by a random generation following a Gaussian distribution.

In more detail, in the NGA, each generation of individuals will be improved with the following strategy:

1. A prescribed vector, $\vec{s}_l$, of selected individuals will be stored.

2. Each of these $\vec{s}_l$ individuals, will breed n_l new temporary individuals with a similar genotype. In total $n_l \times m$ new individual will be generated by the following procedure:

   - The first gene of the original individual will generate n_l random values around his value using a normal distribution with mean equal to the value of the gene and standard deviation, σ, obtained from the cost function from (3.11). These new values are the first genes of the n_l new individuals.

     $$\sigma = (1 - J(\vec{\theta}_i)e_{\min}) + J(\vec{\theta}_i)e_{\max} \tag{3.11}$$

     where e_{min} bounds the minimal value for the standard deviation and $(e_{max} - e_{min})$ give its the decremental value slope (both depend on the problem and search domain).

   - The procedure is repeated for all the remaining genes of the n_l individuals.

3. For each of the n_l temporary individuals originated from each of the $\vec{s}_l$ individuals selected in 1, the cost function will be evaluated. The fittest out of this n_l individuals will be compared with the original individual that generated them. If it is fitter, it will replace it. If not, the original individual is kept in the population.

Figure 3.15 illustrates the local random search strategy for the particular case of $\vec{s}_l = [1]$.

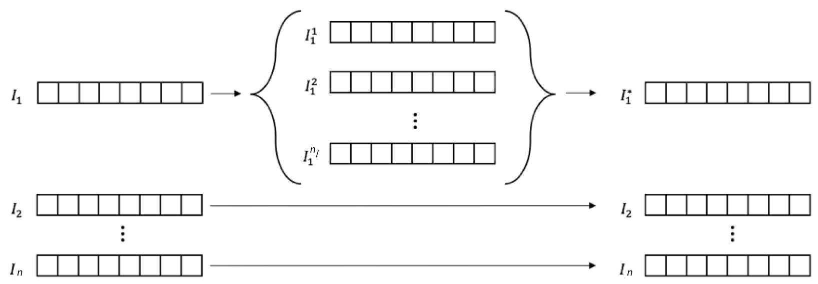


Figure 3.15: Local search strategy for the i^{th} generation for the case of $n_l = 1$.

With the strategy described above, in each generation, the $\vec{s}_l$ individuals may be replaced by fitter individuals. This replacement does not result from mutation or cross-over. It is the outcome of a local random search strategy in the neighborhood of the original individuals.

For the purpose of this work, along with the optimization process, a comparison between methods will be performed. So, for distinction, GA will be treated as Basic Genetic Algorithms, BGA, and for consistency among the methods the conditions to run the GA and NGA optimizations will be,

- Population dimension of 100 subjects;
- Mutation probability of 80%;
- Crossover probability of 30%;
- Stop criteria of 10 000 iterations or by a cost function value of 10^{-10} (in order to force convergence by number of iterations);

Relatively to the NGA two cases will be tested, $\vec{s_l} = [1]$ (NGA_1) and $\vec{s_l} = [1,2]$ (NGA_2). In both cases, a $n_l = 100$ samples per subject will be generated using a normal distribution with the mean of the parameter and standard deviation obtain according to (3.11) and with,

$$e_{\max} = 0.01 \tag{3.12}$$

$$e_{\min} = 10^{-3} \tag{3.13}$$

To compare the performance between BGA and both NGA_1 and NGA_2, 100 runs will be performed in the same conditions.

3.4 Validation and Comparison

The estimation of parameters of any AD model in general, and of the Campestrini et al. [5] model in particular (Eq. (2.20)), is a multidimensional, multimodal, hard to solve problem. For these reasons it is not a good choice for validating and testing the optimization algorithms implemented in this work. Instead, test functions having two decision variables so as to provide visual understanding of the problems, are used to validate and compare the two GA implementations. Two different test functions were considered: the Goldstein-Price function 3.4.1 and the Eggholder function 3.4.2.
At last, in section 3.4.3 it is also developed an IOP using Campestrini et al. [5] model where is estimated the k_6 parameter. With this, it is intended to have a closer code implementation to the problem avoiding the dimensionality difficulties.

3.4.1 Goldstein-Price Function

The Goldstein-Price function it is a good starting point since, although not being concave and unimodal, its global minimum is not difficult to read.

The Goldstein-Price function is given by the expression [39],

$$\begin{aligned} f(p_0, p_1) = &(1 + (p_0 + p_1 + 1)^2(19 - 14p_1 + 3p_0^2 - 14p_1 + 6p_0p_1 + 3p_1^2))(30 + (2p_0 - 3p_1)^2 \\ &(18 - 32p_0 + 12p_0^2 + 48p_1 - 36p_0p_1 + 27p_1^2)), \qquad \forall p_i \in [-2, 2], i = 0, 1 \end{aligned} \tag{3.14}$$

and its global minimum can be found in,

$$f(0, -1) = 3 \tag{3.15}$$

Since this is not an IOP, there are some differences in the way the data is handled. In the first place, the cost function values are not contained between zero and one, which brings a problem to the way the neighbors are generated in NGA. To solve this, the cost function used in this situation is the Goldstein-Price function subtracted by the minimum in order to place the optimum with value zero. Furthermore, the cost function value in expression (3.16) will be divided by 100. This way, a better scaled standard deviation near the solution is obtained.

$$J(\vec{p}) = f(\vec{p}) - 3 \tag{3.16}$$

Moreover, the parameter normalization is performed using,

$$np_i = \frac{p_i + 2}{2 - (-2)} \tag{3.17}$$

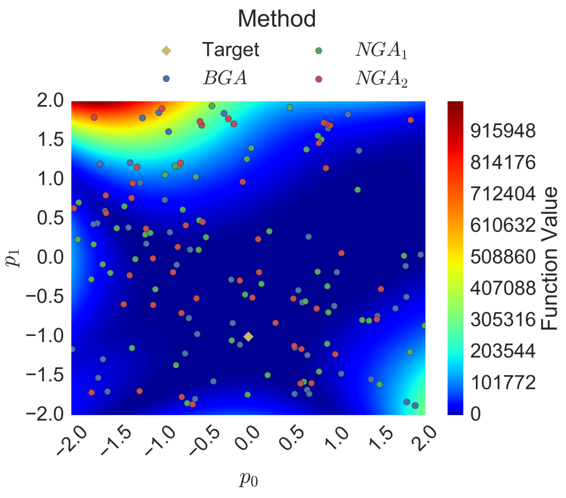
where p_i is the real value of the variable i and np_i stands for the respective normalized one.
The common set of parameters to BGA and NGA is:

- population size of 50 individuals
- maximum iterations of 100
- crossing probability of 0.8
- mutation probability of 0.3

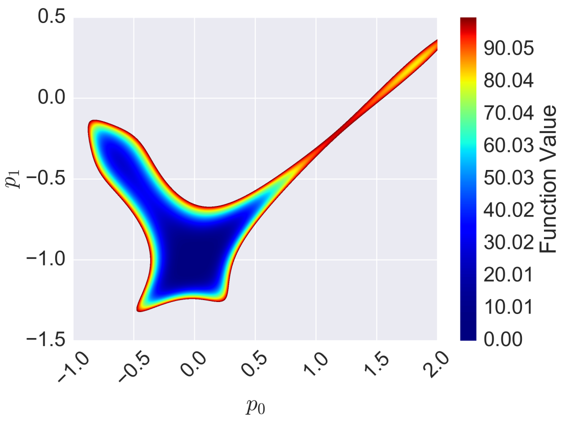
Furthermore, NGA will make use of 50 more intermediate individuals per subject. In order to have a comparison term, 100 optimization runs are also performed. This way is possible to achieve data for a statistical study.

Figure 3.16a shows the initial population, for one run, of BGA, NGA_1, NGA_2 and the optimal point placed over the Goldstein-Price function's plot.

Despite this function be apparently convex, there are three local minima and one global that may create difficulties to optimization process. Performing a refined look to the function in figure 3.16b.



(a)

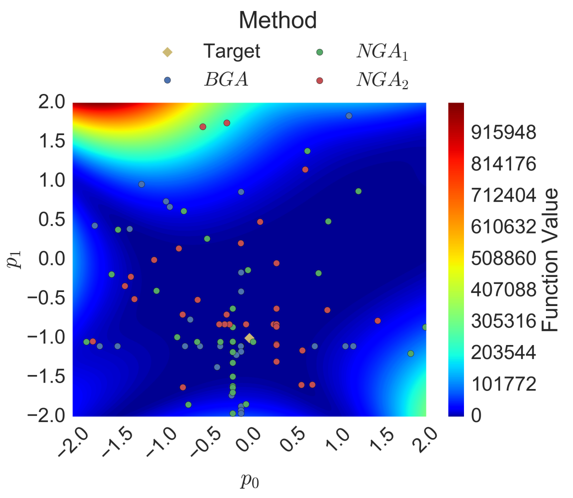


(b)

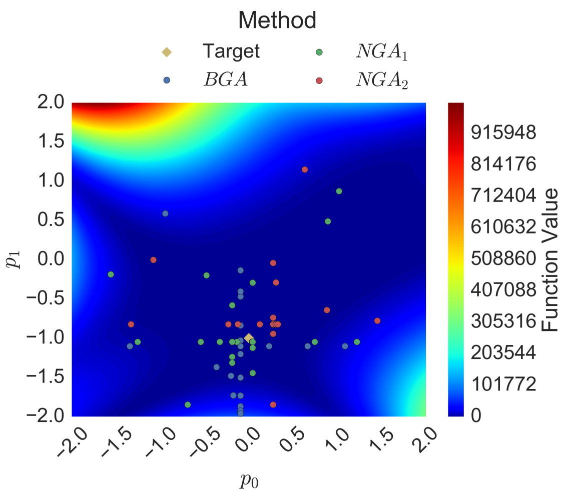
Figure 3.16: In (a) is presented the Goldstein-Price function with an example of initial population of BGA, NGA_1 and NGA_2 and the global minimum. In (b) the function is refined in order to show stagnation points besides the global minimum.

Since data is randomly generated, its dispersion is big and occupies almost all search space. This scatter will be useful for detection of local minimums and, hopefully, the global, as well.

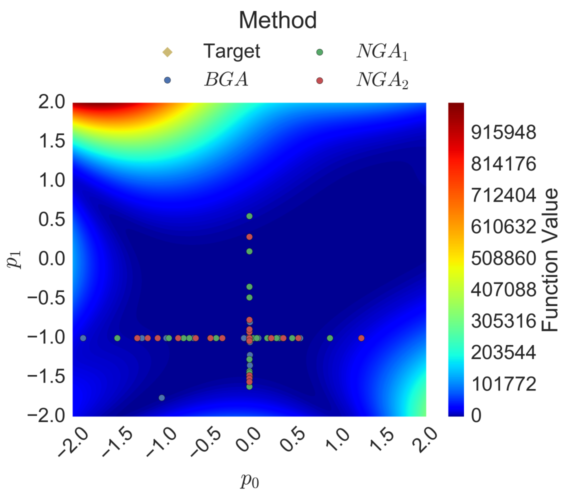
In this particular run two BGA points and one NGA_1 are already near the global minimum. From this initial positions GA operations will be performed on the population in order to converge to the minimum. Figure 3.17 shows four of the iterations obtained for this run.



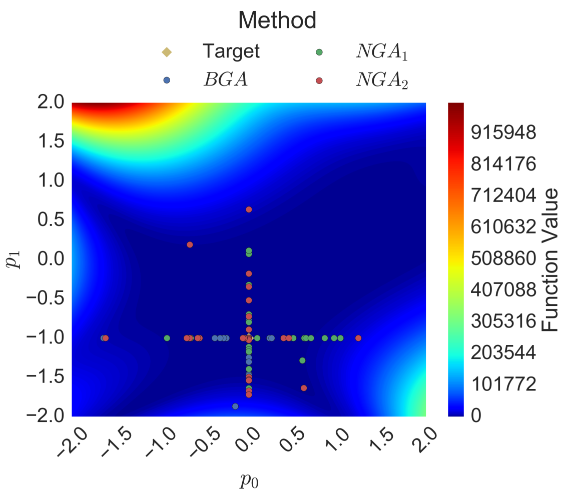
(a) 1st iteration.



(b) 2nd iteration.



(c) 50th iteration.



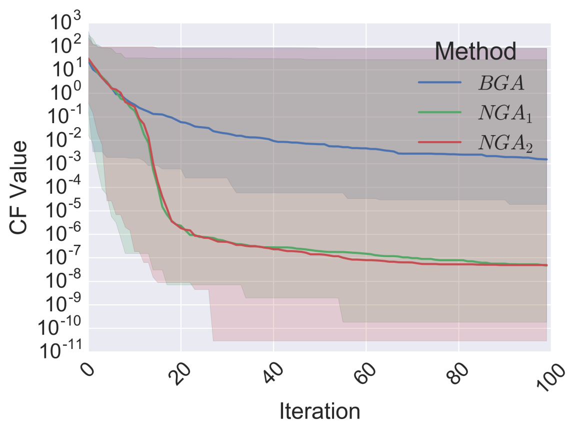
(d) 100th iteration

Figure 3.17: Population dispersion of BGA, NGA_1 and NGA_2 with Goldstein-Price function for 4 iterations.

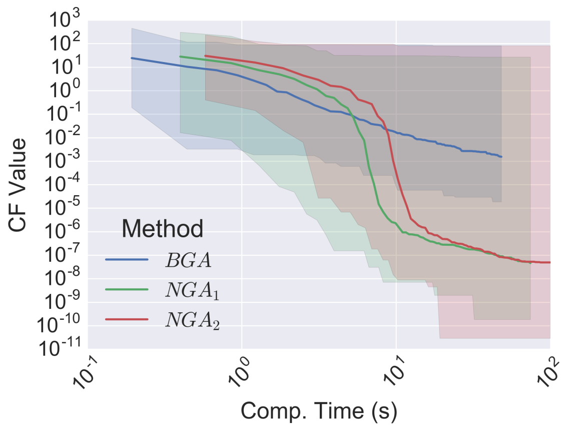
On the first iterations the population rapidly clusters in a cross shape way. This occurs since the cross-over operation only occurs for a single allele, unless mutation also occurs.

In this particular run, all methods were able to find the global minimum, nevertheless from the tendency shown, search did not stop and, for example, in the last iteration points far from the solution may be seen.

The function value versus the number of iterations and the computational time is shown in figure 3.18a and 3.18b, respectively. Since, the computational time per run and iteration is different, the average computational time per iteration for all runs was calculated. The central line, shown, for each method represents the median function value of the 100 runs.



(a)



(b)

Figure 3.18: Comparison of the convergence by iterations and computational time.

The figure 3.18a show that in less than 20 iterations both NGA methods reaches function values smaller than the one obtained by BGA at 100 iterations. For example, if tolerance would be of 0.01, both NGA would already stop.

Table 3.1 presents another comparison of the performance of the GA and NGA methods for the optimization of the Goldstein-price function with a specified accuracy. It shows the percentage of runs that were able to reach a function value of 0.001, and for those runs and accuracy, the average number of iterations and the average computational time needed.

Table 3.1: Computational effort if function's tolerance would be 0.001.

Method	$Iteration$	$Comp. Time$ (s)	$Sucess$
BGA	56.0	20.8	38.0 %
NGA_1	19.8	10.5	99.0 %
NGA_2	18.8	18.8	99.0 %

Although both NGA methods spend more time by iteration (figure 3.19a), the time required to achieve lower function values are also lower, suggesting them to be a viable improvement to BGA method (3.1).

Furthermore, figure 3.18b shows a much better performance where both NGA overcomes the best BGA function value in a less computational time.

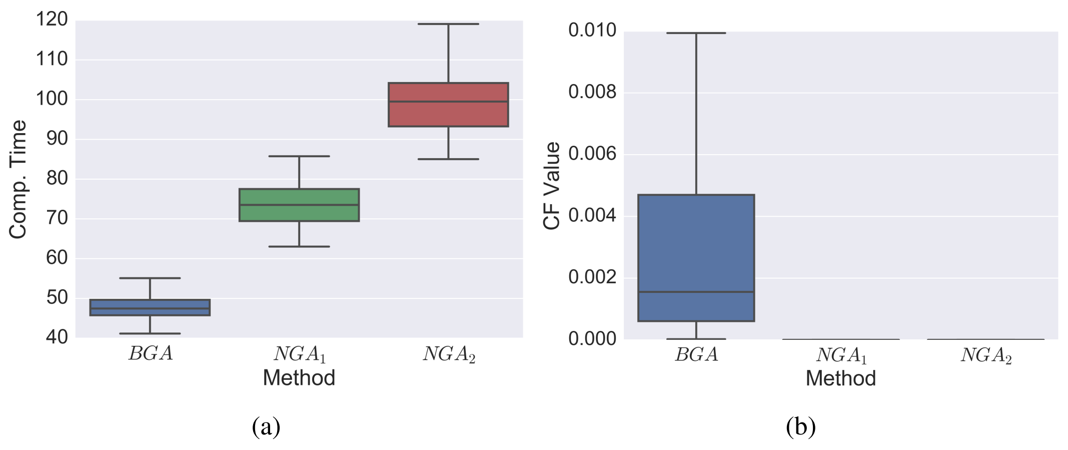


Figure 3.19: Comparison of computational time and function value obtained from the three methods in the last iteration.

The computational time and function value in last iteration are shown in figure 3.19. In both representations, outliers from the plots (BGA - 4 outliers, NGA_1 - 1 outlier and NGA_2 - 1 outlier) were excluded.

From this graphics it is clear that the computational cost increases with the number of subjects used in NGA methods. On the other hand, the consistency achieved with this methods is proven with function value presented after 100 iterations in figure 3.19b.

Accordance to this conclusion are the parameters values (figure 3.20).

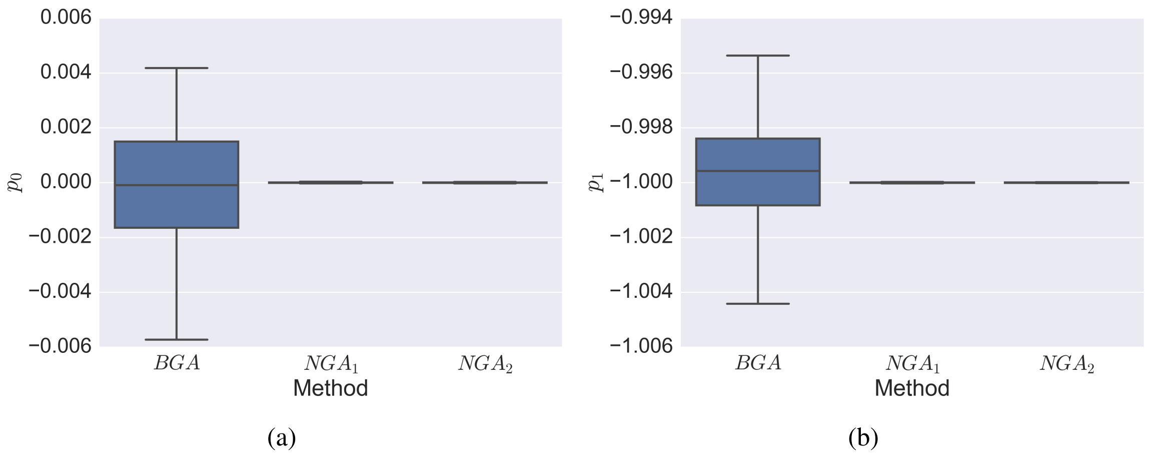


Figure 3.20: Boxplot of parameters obtained after 100 iterations for BGA, NGA_1 and NGA_2 methods.

It is shown that almost all optimizations performed were able to achieve the global minimum ($p_0 = 0$, $p_1 = -1$). Moreover, BGA dispersion is much wider when compared with the NGA results. From this plots, it is shown that, when NGA is capable of achieving the optimum, results are much better.

Figure 3.21a shows a parallel coordinate plot of parameters obtained from 100 optimization runs. This graphic is useful to link variables in multidimensional data. In the horizontal axis is presented the variables in study and, in the vertical axis is shown its value. Variables are then connected by lines that links the obtained results for each run. This technique helps in clusters visualization.

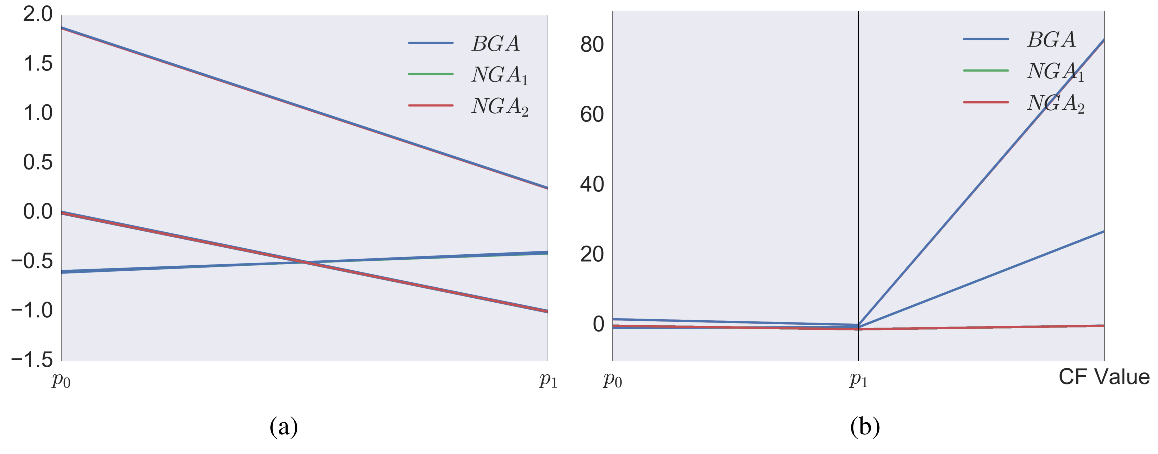


Figure 3.21: Combination of parameters and function value for where GA converged.

The previous figure shows that parameters are consistently achieved, nevertheless there are situations for such optimization has not converged and where parameters has grouped for $p_0 \approx 1.8$ and $p_1 \approx 0.2$ and for $p_0 \approx -0.6$ and $p_1 \approx -0.4$. For those cases, function value (figure 3.21b) are the previous higher and are linked to the outliers obtained (table 3.2). Moreover, those are 2 of the 3 local minima of this function.

Table 3.2: Outliers obtained from the 100 optimizations performed.

Method	p_0	p_1	CF Value
BGA	-0.605	-0.394	27.016
BGA	-0.593	-0.409	27.035
BGA	1.878	0.252	82.019
BGA	-0.590	-0.408	27.040
NGA_1	-0.590	-0.412	27.056
NGA_2	1.870	0.245	81.825

What can be seen is that BGA presents more outliers than any of the NGA methods proving more consistency in the results obtained from the improved GA.

3.4.2 Eggholder's Function

Another test function that was used to validate and compare the different GA implementations is the Eggholder function. This is a function with many valleys which implies many minimums, moreover, some of those are very steep. In this test consistency and the ability of GA to unlock from a local minimum and perform search in others zones of the function will be searched.

According to [40], Eggholder's function is given by,

$$f(\vec{p}) = \sum_{i=1}^{n-1} \left[-p_i sin\left(\sqrt{|p_i - p_{i+1} - 47|}\right) - (p_{i+1} + 47)\, sin\left(\sqrt{|0.5p_i + p_{i+1} + 47|}\right)\right], \qquad (3.18)$$

$$\forall p_i \in [-512, 512], i = 1, 2$$

and its minimum location is, for $n = 2$,

$$f(512, 404.2319) = -959.6407 \qquad (3.19)$$

For coding simplicity reasons, function minimum absolute value was added to the function in order to make function always positive with a minimum of zero. Figure 3.22 presents the function for the considered domain with initial GA populations from the three methods of one of the runs. For this optimization, all GA parameters considered in the section 3.4.1 were maintained.

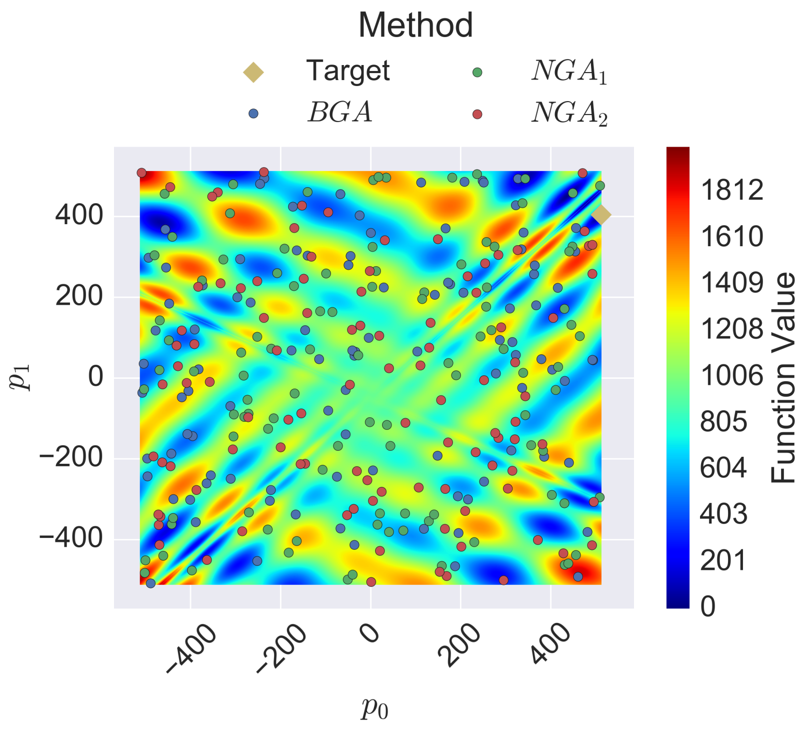
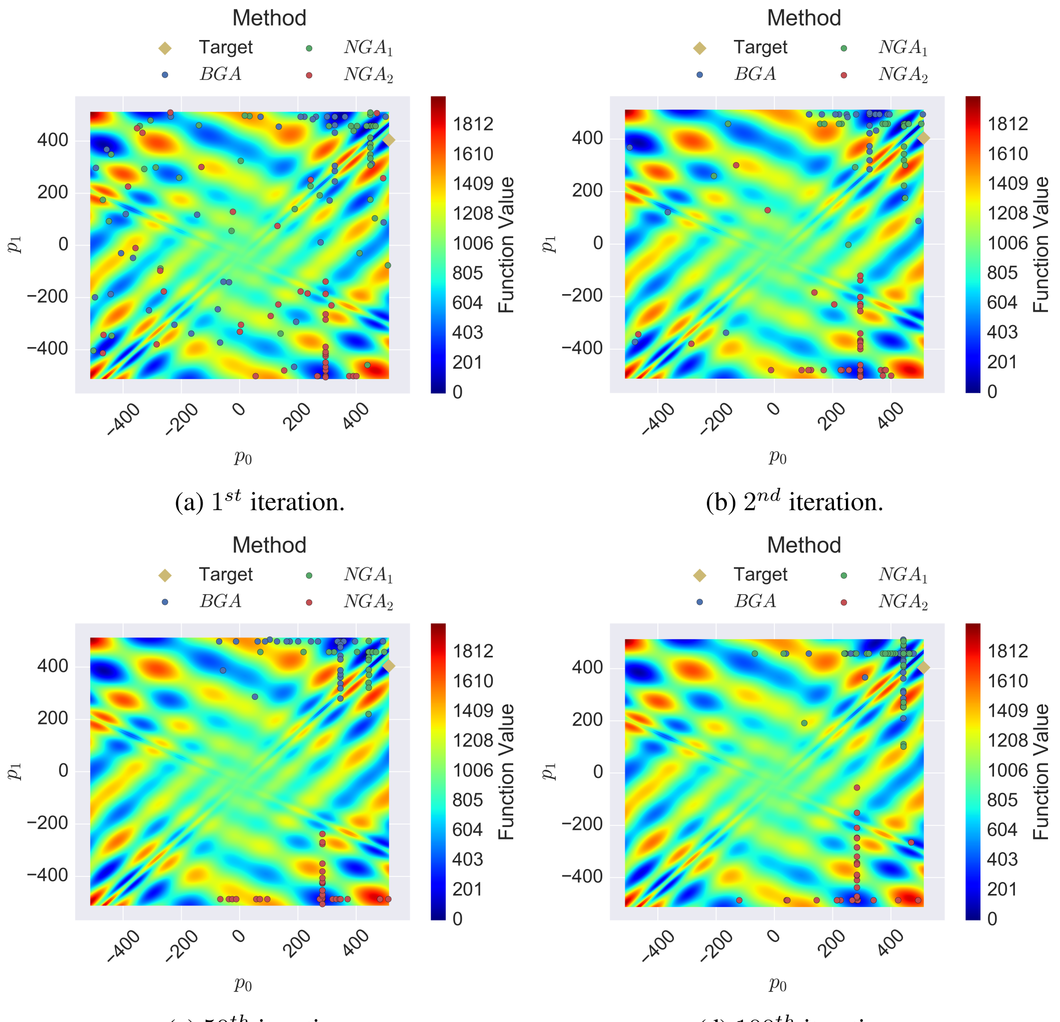


Figure 3.22: Eggholder function with GA initial populations.

Among all points, it can be seen that some are already near the global minimum while others are placed in near local minimums. From this, it is expected that the GA operations results in a clustering of the population near the lower function values of the initial population. This may not be the global minimum and, for that reason, all population may be pushed away of the global optimum. Figure 3.23 shows some iterations from one of the 100 optimization runs performed.



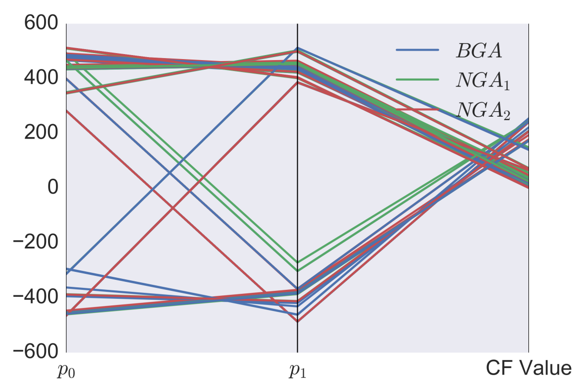
(a) 1st iteration. (b) 2nd iteration.

(c) 50th iteration. (d) 100th iteration

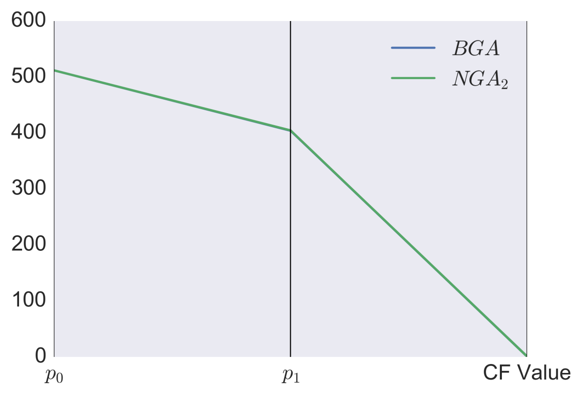
Figure 3.23: Population dispersion in Eggholder function.

It can be seen that the population has clustered around local minima, which resulted in difficulties for the optimization process. This minima have locked the search locally and have increased the difficulty in finding the objective. This is mainly consequence of the choices made for the GA crossing operation. Since each parameter is individually generated by a weighing of to subjects, than population is constrained in a cross formation. The only way for an individual to try other possibilities is if a cross and a mutation occurs at the same time.

Figure 3.24 shows the dispersion of results obtained from all optimization processes (figure 3.24a) and the same data filtered for function values less than 1.0 (figure 3.24b).



(a) From all runs.



(b) From all results with function value less then 1.0.

Figure 3.24: Last iteration of Eggholder's optimization.

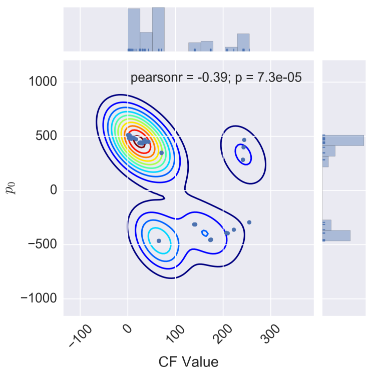
This indicates low consistency in estimating the optimum, nevertheless the greater amount of results are near solution. Despite that, those results are in a near local minima which shows that 100 iterations with the specified GA parameters are not able to converge.
Table 3.3 presents the numerical results that are shown in figure 3.24b.

Table 3.3: Statistical data information of results obtained from the three methods.

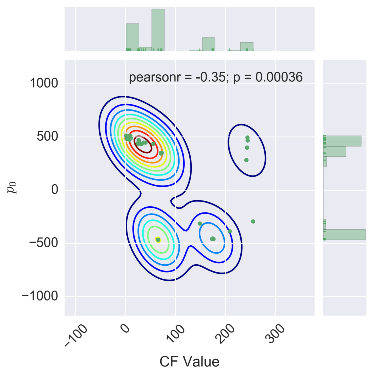
Method	p_0	p_1	CF Value
BGA	511.771	403.885	0.80800
BGA	511.944	404.086	0.20138
NGA_2	511.840	404.220	0.56436
NGA_2	511.849	404.724	0.96721
NGA_2	511.941	404.427	0.27114

Since NGA methods are being used in the first and second individuals, they are always trying to improve the estimation and leaving the global search effort for the typical GA strategy. For that reason, the inability of getting to the global minimum is not being of the responsibility of NGA or BGA distinctions, but a matter of getting luck in the optimization process.
One way to overcome this difficulties may reside in increasing the population dimension and/or the use of NGA method using also individuals that are in the middle of the population ranking.

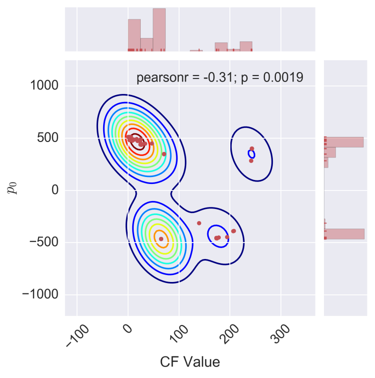
In an attempt of getting a general clustering information, is also presented figures 3.25 and 3.26. They show the scatter of the parameters by function value and, on each axis, is also presented an histogram that is related to each variable of the respective axis. There are also, level curves that represents the intersection of those histograms.
In figure 3.25 is presented the comparison of p_0 with the function value.



(a)



(b)

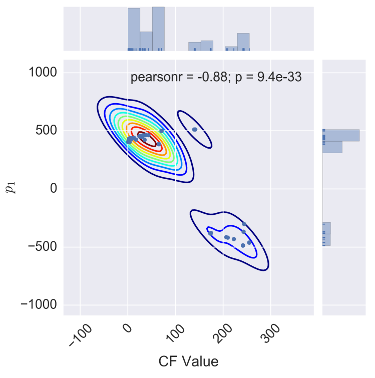


(c)

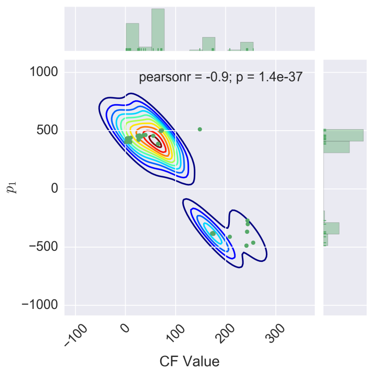
Figure 3.25: Parameter, p_0 dispersion by Eggholder function value for last iteration of all runs for each one of the BGA and NGA methods.

Despite the difficulty experienced by the algorithm in converging to the global minimum, it can be seen from the previous figure that, there is a great amount of occurrences near the solution. This is a good indicative that, with more iterations there are good possibilities of converging to better results. Moreover, from this figure is possible to have an idea of where the minimums are using level curves as reference. Comparing its center with the parameter value and figure 3.22 it can be seen that there is a minimum for those cases.

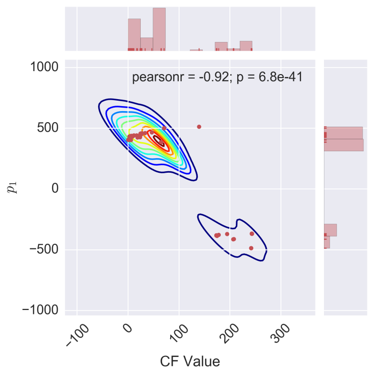
Figure 3.26 presents the comparison of p_1 with the function value of the last iteration of all runs.



(a)



(b)



(c)

Figure 3.26: Parameter, p_1 dispersion by Eggholder function value for last iteration of all runs for each one of the BGA and NGA methods.

Once again, can be found a tendency in data for the target, nevertheless clusters out of the absolute minimum can still be detected.

The problem with this data visualization is that the link between the two parameters are lost, but, on the other hand, it is useful to detect how data is related with the cost function and, helps on identification of identifiability/low sensitivity problems.

3.4.3 k_6 Parameter Optimization of Campestrini's et al. Model

To test Campestrini et al. model (2.20) implementation, in this section, it is performed the estimation of parameter k_6. To achieve this, the accumulated biogas produced (figure 2.10) is used to evaluate the cost function defined in (3.6). Furthermore, for the optimization purposes, except for the number of iterations (maximum of 50 iterations) GA parameters defined in previous sections are kept the same.

In figure 3.27 the cost function is plotted, as a function of the parameter value. In the same plot, the initial GA population for the three methods is also represented.

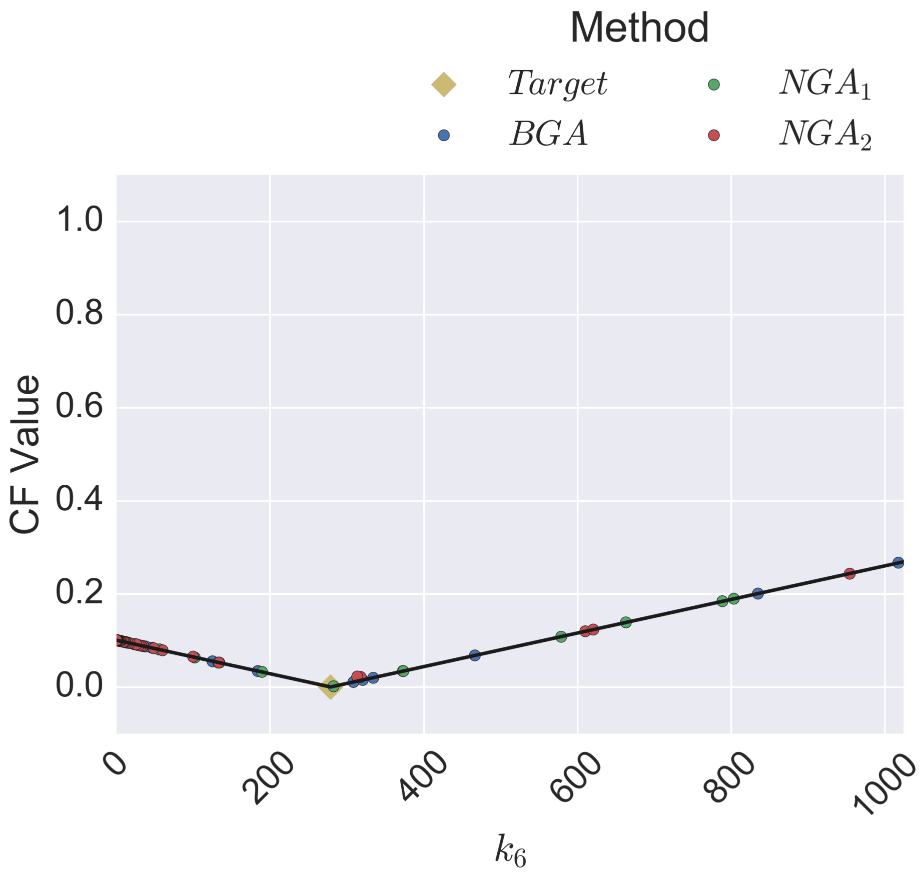
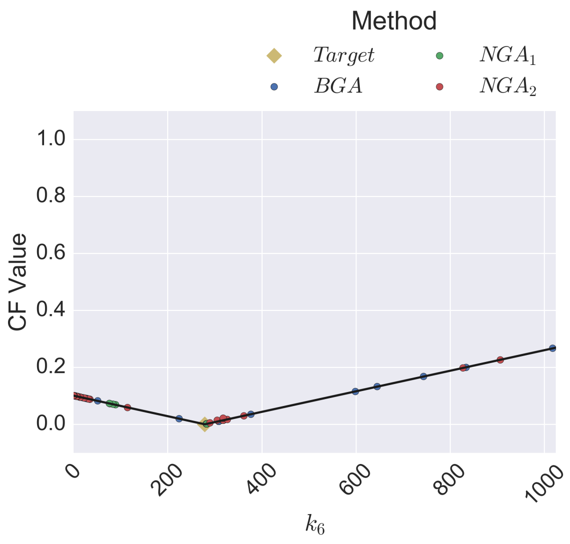


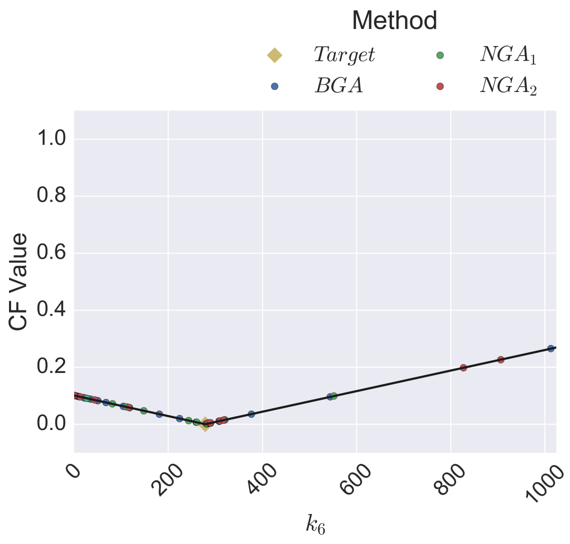
Figure 3.27: Cost function of Campestrini et al. model for k_6 optimization with initial population of BGA, NGA_1 and NGA_2 and the optimal.

This is a convex function with a single minimum, so optimization should be an easy task when compared to the ones before. Nevertheless, this is a good starting point to validate the implementation of optimization methods.

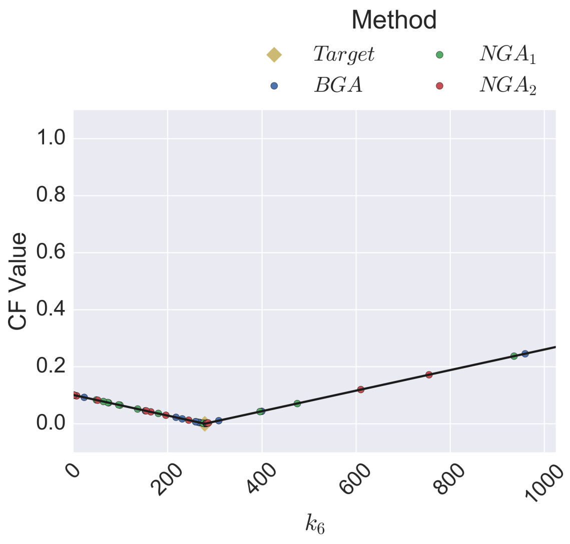
From the 50 iterations, for one of the runs, are presented 4 iteration in figure 3.28.



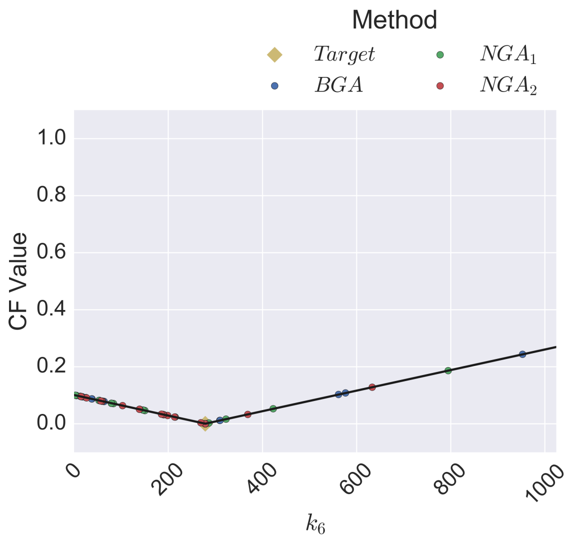
(a) 1^{st} iteration.



(b) 2^{nd} iteration.



(c) 4^{th} iteration.

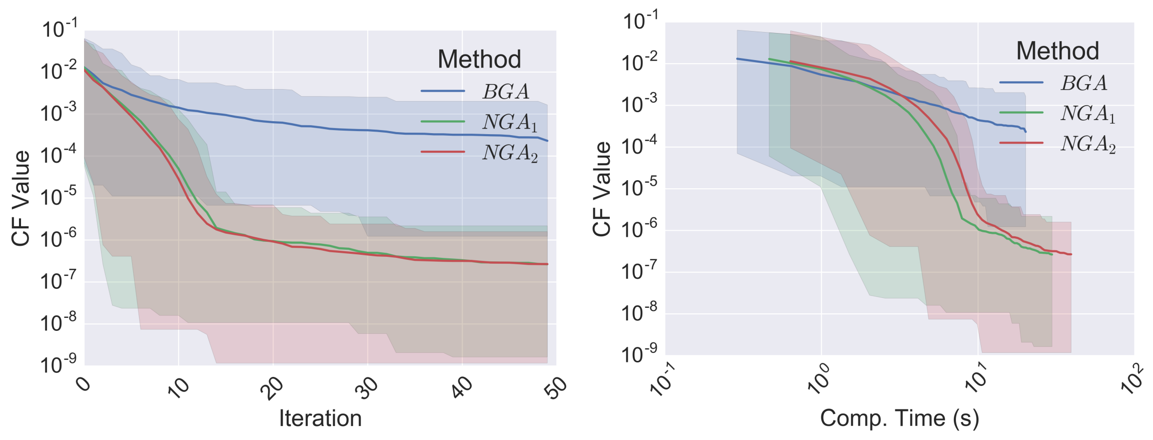


(d) 50^{th} iteration

Figure 3.28: Cost function of Campestrini et al. model for k_6 optimization with population dispersion for four iterations.

Population is initially dispersed, but at the first iteration it is already grouped around the minimum. After that, two things happen, the population that is close to the minimum refines the solution and global search is kept which can be seen on the other iterations with the moving of population along the cost function value.

In figures 3.29 can be found the comparison of cost function value with the iterations (figure 3.29a) and the computational time (figure 3.29b).

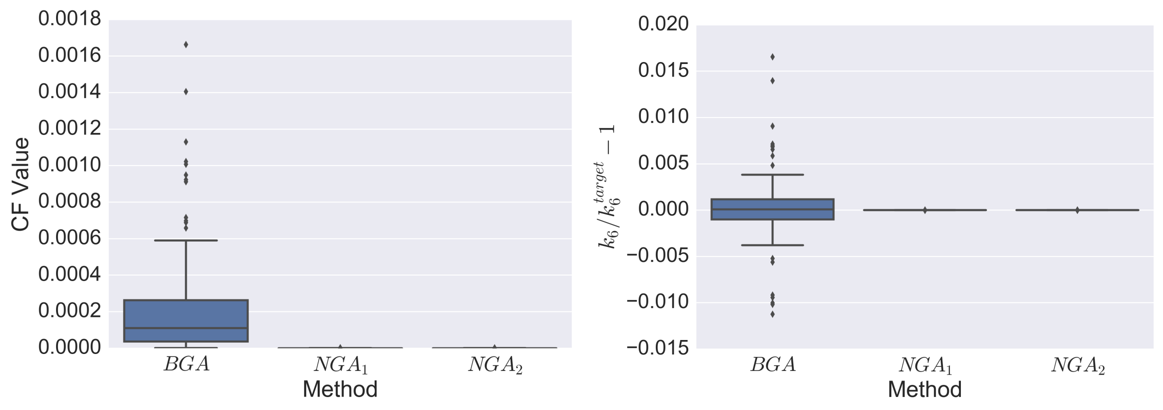


(a) (b)

Figure 3.29: Compare of convergence by iterations and the final function value by method.

If, on one side, less iterations are needed to decrease cost function value (3.29a), by the other, NGA methods achieve lower cost function values after around 5 seconds of computational time (3.29b).

In figure 3.30 a comparison for both final cost function distribution for each method and the estimated parameter are presented.



(a) (b)

Figure 3.30: Comparison of computational time cost and function value obtained for the three methods.

Despite the good results presented by BGA in 3.30a it is also shown the consistence of the NGA method that is able to achieve the exact solution for all the 100 runs performed. Cost function is intrinsically related with the parameter that is being estimated, for that reason figure 3.31 shows a parallel coordinate plot where both parameter and cost function value are close to the solution and to zero, respectively.

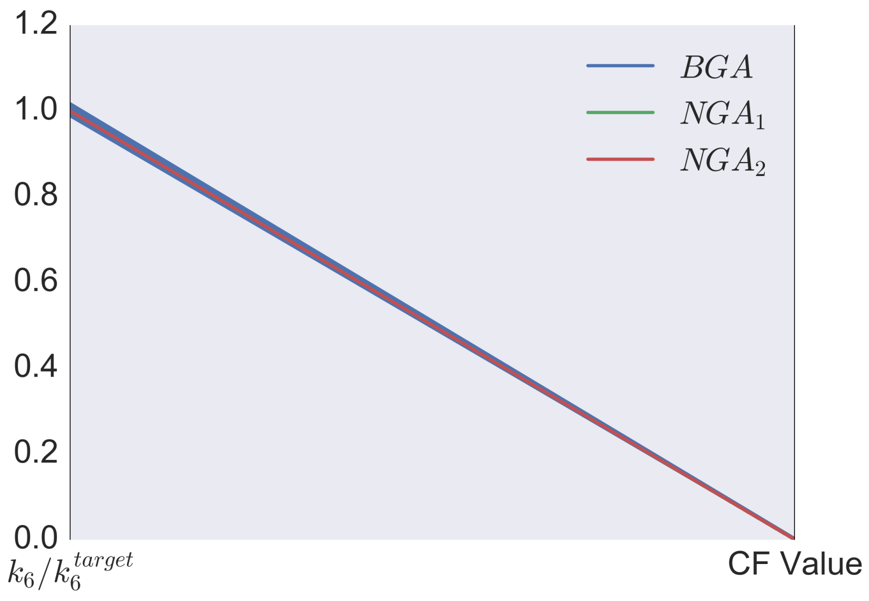


Figure 3.31: Parallel coordinate showing relation between the estimated parameter and the respective cost function value for the 100 runs.

What is also seen from this figure is that the cost function value varies little with the parameter fluctuation (table 3.4). This is indicative of low sensitivity to the parameter by the cost function value which may produce difficulties in the convergence to the solution when more parameters are being estimated.

Table 3.4 shows some statistical information of the estimated parameter and its cost function value obtained from the 100 runs using BGA method.

Table 3.4: Statistical data information of results obtained from BGA method.

	k_6	CF Value
mean	278.669119	0.000232
std	1.100267	0.000332
min	275.487844	0.000001
25%	278.338539	0.000037
50%	278.640792	0.000110
75%	278.942872	0.000263
max	283.233110	0.001664

From previous table stands out the comparison of standard deviation which shows that parameter can vary about a unit producing very little effect on cost function. This is proof of low sensitivity of the parameter when using this cost function.

Moreover, normalizing between the maximum and the minimum for both, k_6 and $CF\ Value$,

$$\frac{\max - \min}{\max} \times 100 = \begin{cases} k_6 = 2.735\% \\ \text{CFValue} = 99.94\% \end{cases}$$

This means that when k_6 has a variation range of around 2.8%, the cost function value will have a variation range of 99.9%. This is a confirmation that this cost function has low sensitivity to the parameter.

Chapter 4

Optimization With Simulated Results

Due to all the difficulties associated with the IOP involved in the calibration of AD models [41] and [42], an optimization with noise-free results was undertaken (section 4.1). These results were presented in section 2.4 and were obtained with the model described in Campestrini et al. [5] with a particular set of initial conditions and parameters. A target accumulated methane flow rate was generated and the objective of the calibration procedure is to find the parameters that are able to reproduce this target. Since the solution to the problem is known beforehand, it is possible to test and critically analyse the optimization methods implemented and developed in this work.
Due to the high dispersion that is still presented in the results obtained from GA, the Nelder-Mead optimization method was used to improve the results. The results obtained from the GA were used as initial guess.

4.1 GA Optimization

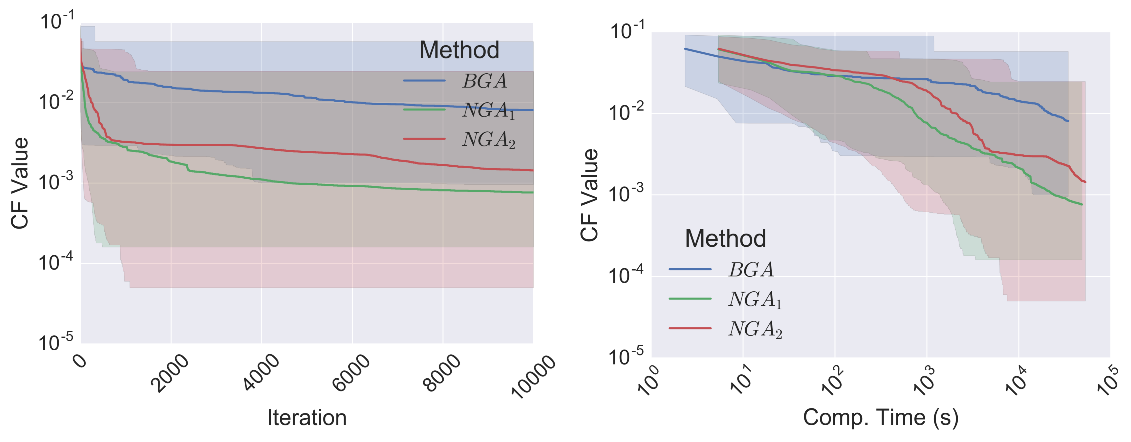
Optimization using GA is performed in order to understand if all parameters are possible to be estimated making use of the following considerations:

- population size of 100 individuals
- maximum of 10 000 iterations
- crossover probability of 0.8
- mutation probability of 0.3

Along 10000 iterations Basic Genetic Algorithms (BGA) and Neighbored Genetic Algorithm (NGA) will be compared according to the defined conditions in section 3.3.

Since GA methods are stochastic, the comparison of methods must be based on result from statistical analyses. For those purpose the three methods were run 100 times.

According to those 100 runs, figure 4.1, presents cost function values versus the number of iterations performed (figure 4.1a) and versus the computational time spent (figure 4.1b).The graphic shows a shaded area that delimits the maximum and the minimum cost function values for the 100 runs and a line with its median.



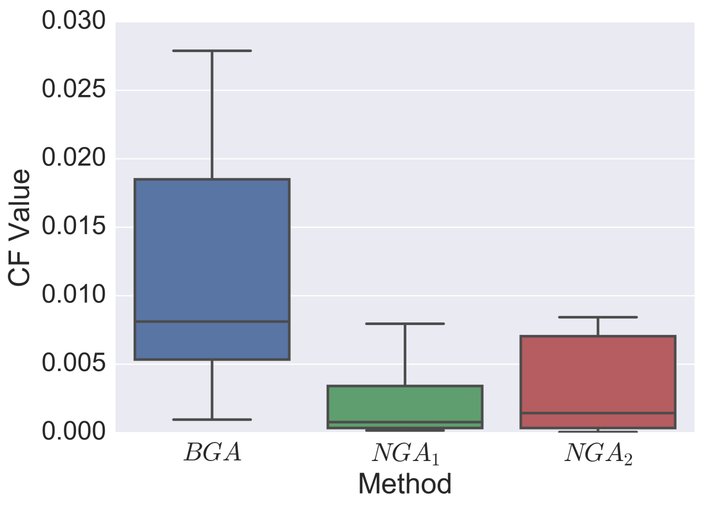
(a) (b)

Figure 4.1: Comparison of convergence for BGA, NGA_1 and NGA_2 methods; In graphic 4.1a it is shown the cost function value vs iteration and 4.1b the cost function value vs CPU time.

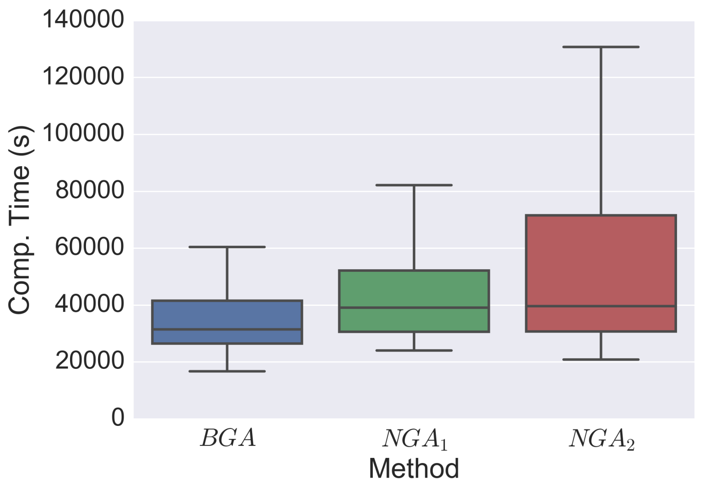
It stands out that both NGA methods overcome the BGA method in performance. If on one side after 10000 iterations BGA has a cost function value little less than 10^{-2}, on the other after less than 300 iterations both NGA methods have achieved the same cost function value continuing to decrease at a high rate.

In figure 4.1b it is shown that at the end, NGA methods, despite spending more computational time per iteration, can still return lower cost function values more often in less time.

Figure 4.2a shows the cost function values obtained in the last iteration of the 100 runs. A boxplot where the CPU time spent in all runs and methods is also shown (figure 4.2b).



(a)



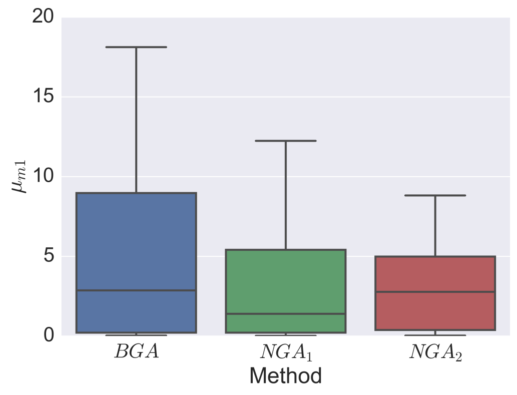
(b)

Figure 4.2: Comparison of cost function dispersion and computational time for BGA, NGA_1 and NGA_2 methods from all runs excluding the representation of outliers.

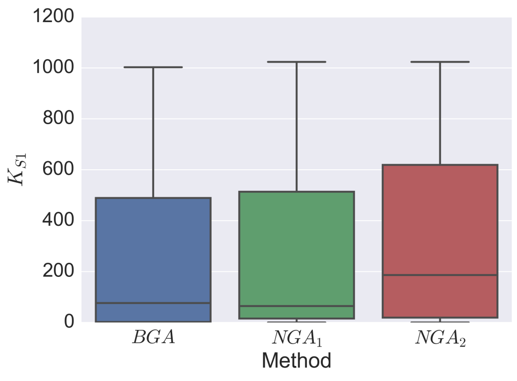
Apparently, NGA_1 has been the one that has converged to lower cost function values more often followed by the NGA_2. As in the case of the Goldstein-Price function no improvement was achieved when using 2 individuals instead of 1, for the local search. In both NGA methods, whisker's end stay below the median value of BGA confirming the better performance suggested by figure 4.1.

Unexpectedly, the median values of the computational time of the two NGA implementations are around the same, even though the NGA_2 results present more dispersion.

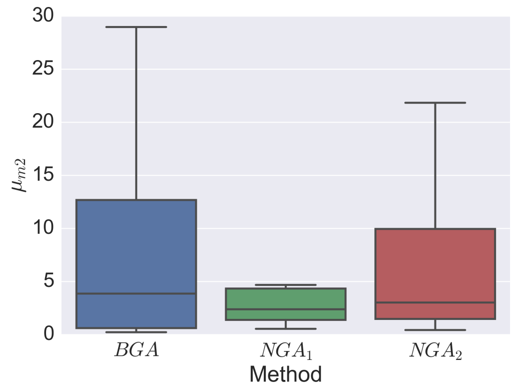
It is possible to see the distribution of each parameter estimated for all runs in figure 4.3. Despite existing outliers are not shown the boxplots.



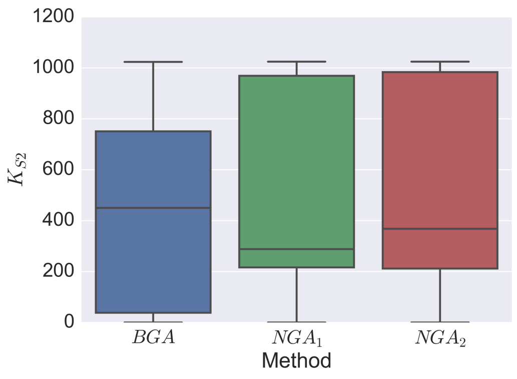
(a) $\mu m1^{\text{targ}} = 0.42912$ day^{-1}



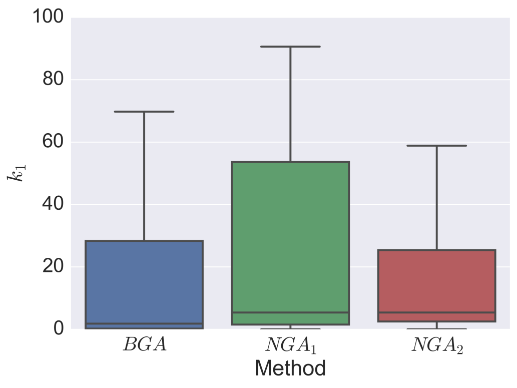
(b) $K_{S1}^{\text{targ}} = 13.065$ mg/L



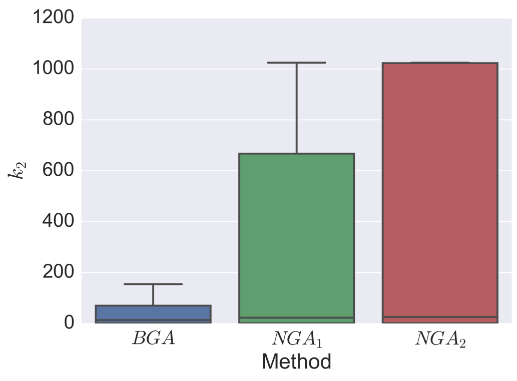
(c) $\mu m2^{\text{targ}} = 2.6943$ day^{-1}



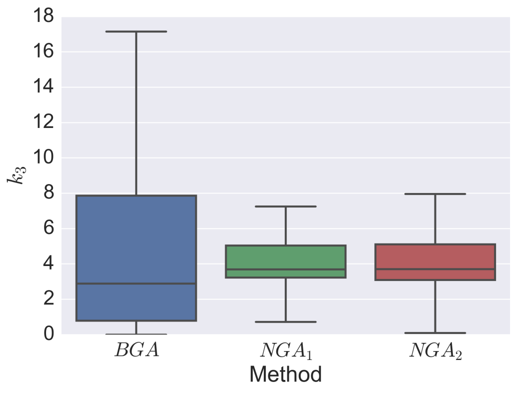
(d) $K_{S2}^{\text{targ}} = 571.27$ mg/L



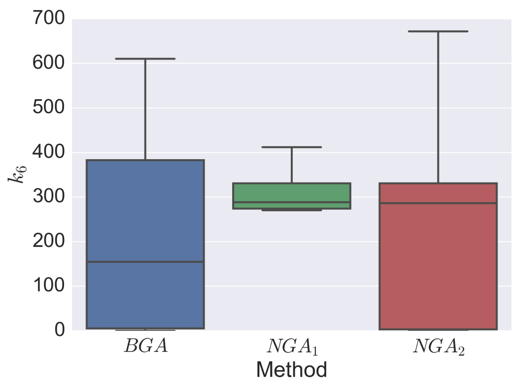
(e) $k_1^{\text{targ}} = 0.31204$ mg COD/mg x_1



(f) $k_2^{\text{targ}} = 0.062776$ mmol VFA/mg x_1



(g) $k_3^{\text{targ}} = 0.062776$ mmol VFA/mg x_1



(h) $k6^{\text{targ}} = 278.62$ mg CH_4/mg

Figure 4.3: Comparison of parameter estimation for BGA, NGA_1 and NGA_2 methods from the last iteration of all runs.

From figure 4.3 it is possible to conclude that generally, there is a wide dispersion in the solutions of the parameters obtained for this calibration problem. If, for one side, NGA_1 shows better consistency in estimating μ_{m2}, on the other side, it is notable that, for example, in the case of k_2, BGA, shows an apparent better result than both NGA methods. In particular, k_2 presents, for the NGA_2 case, a dispersion that occupies all the search domain.

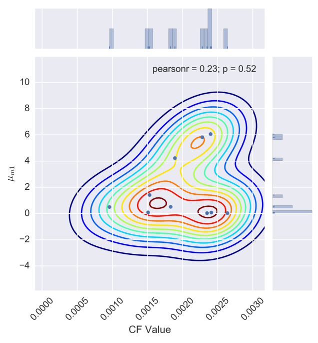
What can be concluded from the previous figures is that, at least, there is low sensitivity of cost function relatively to parameters.

As shown before, there were multiple runs where incorrectly fitted parameters were obtained. Because those results were interfering with the understanding of the overall analysis, filtering of those data was performed. The way it was done was by excluding all parameter values that are outliers and results with cost function values over 0.003. Figure 4.1 is useful to analyse the convergence rate, but there is not guarantee that low cost function values mean a good estimation for the parameters. Identifiability of one or more parameters may not exist for the chosen domain, for example.

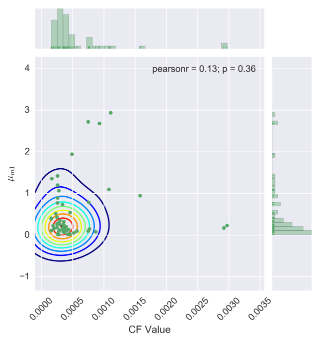
Figures 4.4 and 4.5 show scatter plots of the cost function and respective parameter values. Since GAs, usually cluster the population around the minimum, this charts may also be useful to identify the presence of local minima and, in case of no-identifiability, other global ones. For that purpose, a distribution at each axis and level curves are also shown in order to better visualize those clusters.

Even after filtering, parameters estimates present very dispersed values. This could be a consequence of the wide search domain, a non-identifiability indicator for some parameters, low sensitivity or, possibly, not enough iterations.

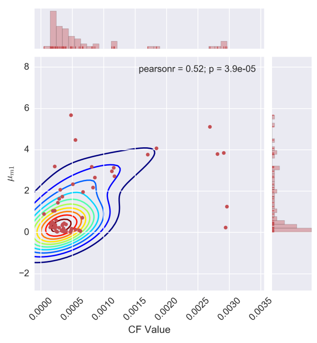
The previous figures also points the need to take care with the relationship between cost function values and parameter values, since situations exist, where lower cost function values correspond to worse parameter results. From this, having reached convergence does not necessarily imply a direct relationship between cost function and the parameter value since there is dispersion even for low cost function values.



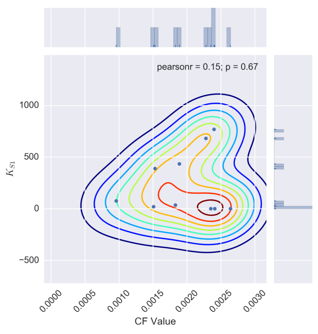
(a) BGA.



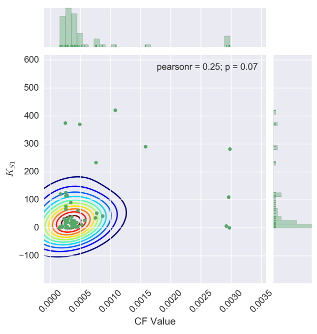
(b) NGA_1.



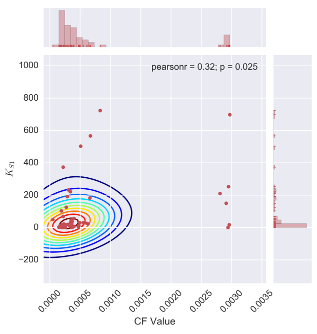
(c) NGA_2.



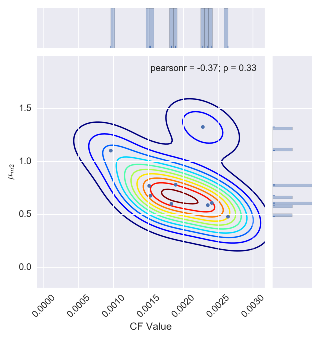
(d) BGA.



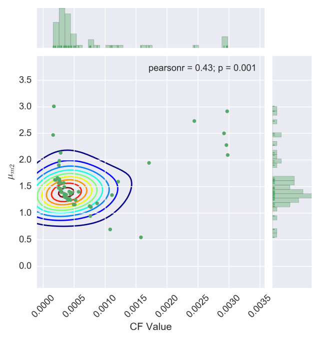
(e) NGA_1.



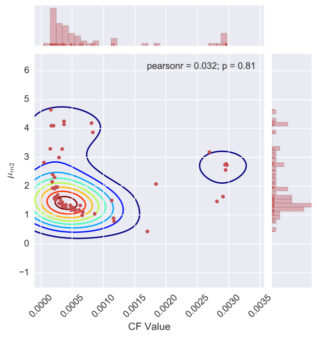
(f) NGA_2.



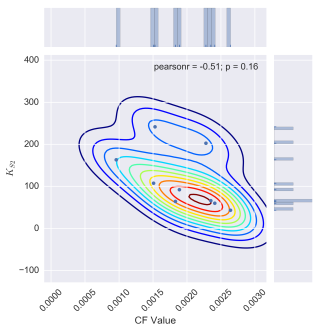
(g) BGA.



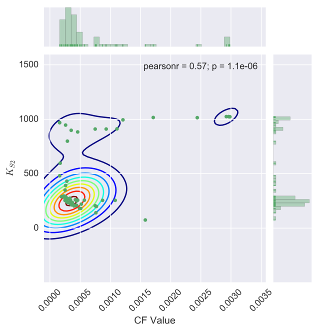
(h) NGA_1.



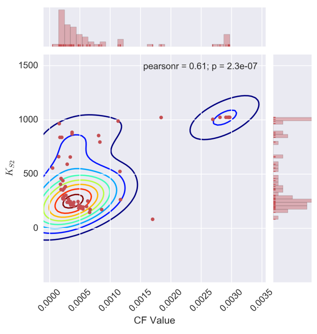
(i) NGA_2.



(j) BGA.

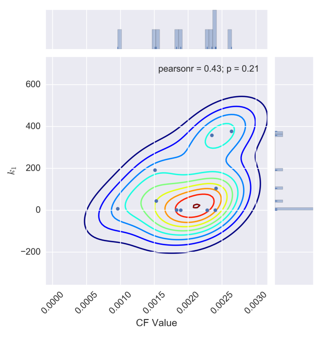


(k) NGA_1.

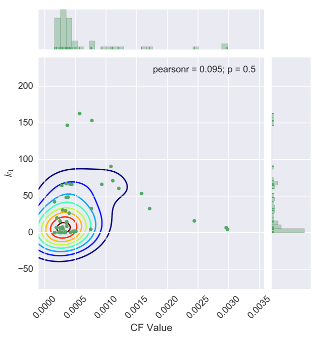


(l) NGA_2.

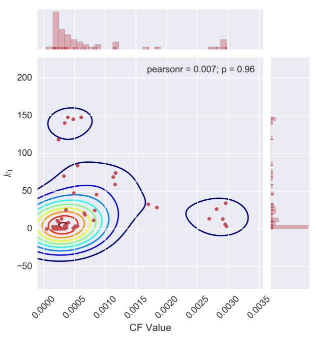
Figure 4.4: Cost function values versus parameter estimates for all runs filtered by cost function values and with no outliers.



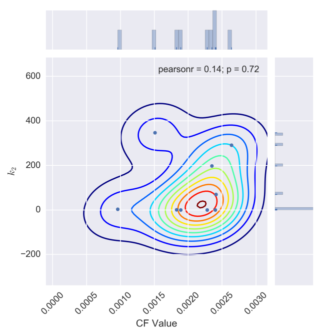
(a) BGA.



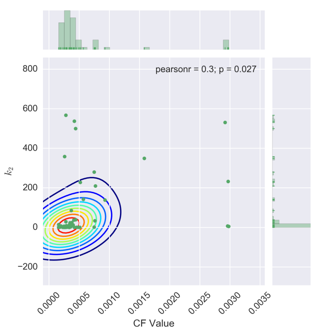
(b) NGA_1.



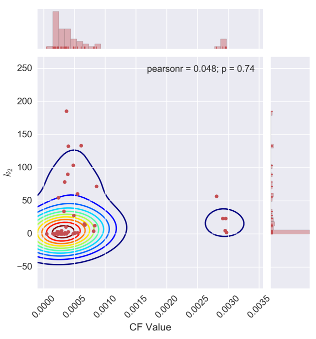
(c) NGA_2.



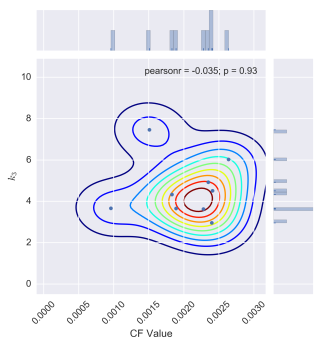
(d) BGA.



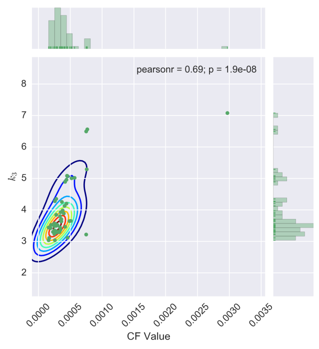
(e) NGA_1.



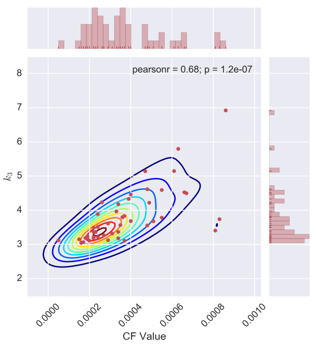
(f) NGA_2.



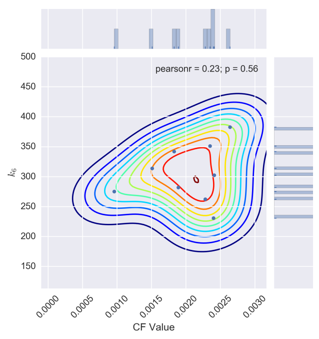
(g) BGA.



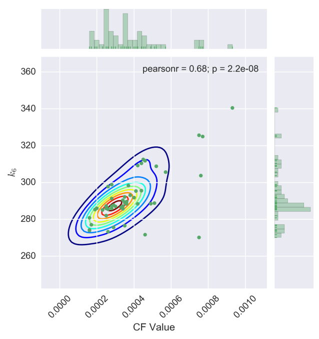
(h) NGA_1.



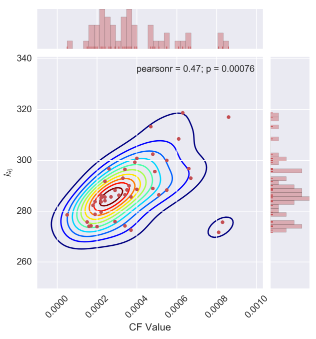
(i) NGA_2.



(j) BGA.



(k) NGA_1.



(l) NGA_2.

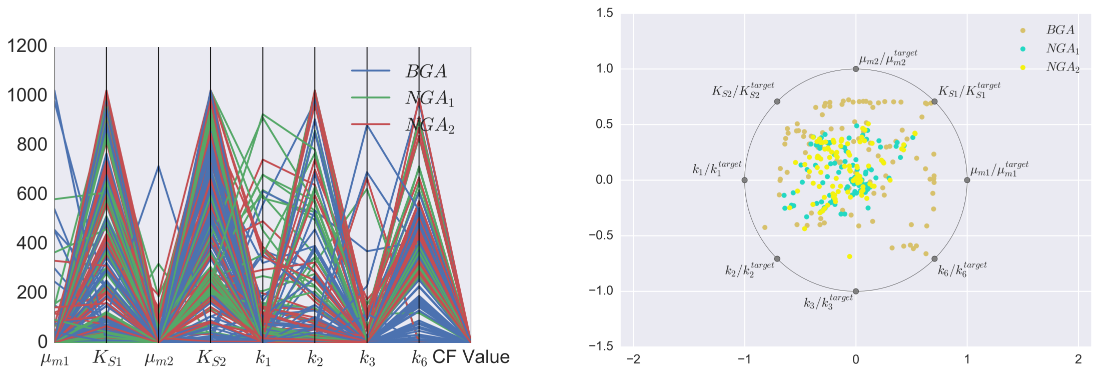
Figure 4.5: Cost function values versus parameter estimates for all runs filtered by cost function values and with no outliers.

It is also visible that for all parameters, after filtering the amount of data is much higher for the 2 NGA implementation then for BGA. Additionally, the cost function values of NGA values are much lower when compared with the BGA. This is an indication of the NGA consistency.

Despite the wide dispersion, in most cases, the clustering shows some tendency to the solution with the decrease of the cost function value.

From all the previous analyses, one can conclude that for an experimental situation, where parameters are completely unknown, it would be easy to calibrate the AD model for "wrong" parameters, since there is a great amount of data that are far from the ones desired. Despite the previous conclusions, more runs would be useful to confirm.

Since from the previous analyses it is not possible to relate each of the estimated parameters, in figure 4.6 it is shown a parallel coordinate plot (figure 4.6a) and a Radviz plot (figure 4.6b). This last graphic is a way to visualize multidimensional data and to gain sensibility to the influence that each parameter has over the others. The typical example is to think that each parameter has a spring from the center to the data element and the greater the influence a parameter has the greater is the force and displacement of that element to the parameter.



(a) (b)

Figure 4.6: Results from the last iteration of all runs using multidimensional visualization techniques.

What can be seen from figure 4.6a is the confirmation that the majority of the runs has resulted in a wide dispersion in all estimatives. Moreover, it is possible to say, from that figure, that K_{S1}, K_{S2}, k_2 and k_6 are the ones that shows lower consistency. In figure 4.6b the parameters that shows greater capability to push data from the center are those same parameters.

In an attempt to find if there are some runs that were capable of achieving the correct estimative of the parameters, a lower filter of 0.0001 is applied to the cost function (figure 4.7).

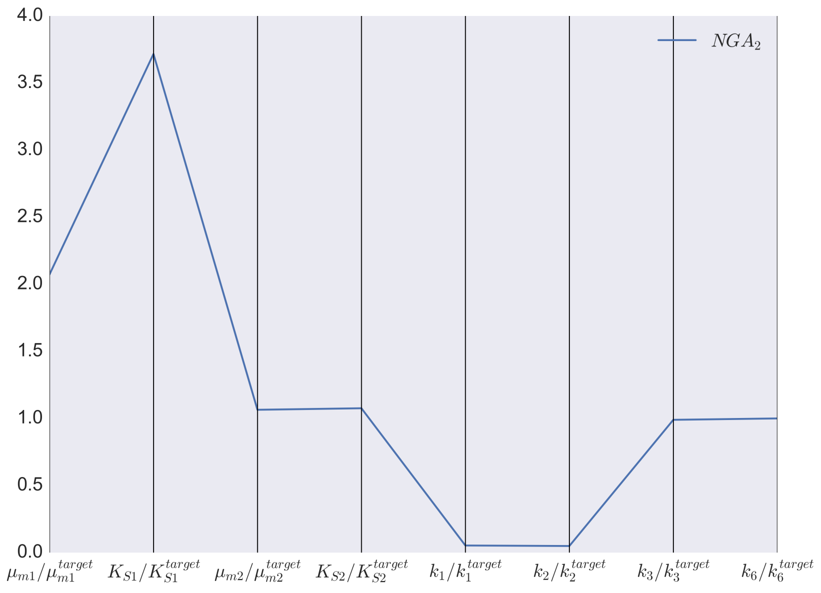


Figure 4.7: Last iteration of the best run.

From the previous figure it is possible to see that only one of the runs was able to achieve a cost function value of 0.0001. Moreover, the only method that was capable of achieving this was the NGA_2. Nevertheless, the target parameters were not reached and there is one parameter almost 4 times higher from the correct one.

Table 4.1 shows the parameter values obtained when the cost function value has a value of 5.0×10^{-5} that is to say of the best estimative from all runs.

Table 4.1: Last iteration of the best run.

μ_{m1}	K_{S1}	μ_{m2}	K_{S2}	k_1	k_2	k_3	k_6
0.88952	48.55571	2.81982	556.43226	0.01645	0.00307	3.11435	278.56934
μ_{m1}^{target}	K_{S1}^{target}	μ_{m2}^{target}	K_{S2}^{target}	k_1^{target}	k_2^{target}	k_3^{target}	k_6^{target}
0.42912	13.065	2.6493	571.27	0.31204	0.062776	3.1473	278.62

4.2 Nelder-Mead Optimization

In order to try to decrease the cost function value even further, an optimization using the Nelder-Mead algorithm is performed having as initial guess the results obtained from GA. Starting from results that should be close to the optimized solution, it is expected that the Nelder-Mead algorithm is able to converge to the correct parameter values more easily. The stopping criteria for the Nelder-Mead algorithm was by reaching 5000 iterations. Figure 4.8 shows the cost function value after convergence for the Nelder-Mead algorithm grouped according to the algorithm that generated the initial guess.

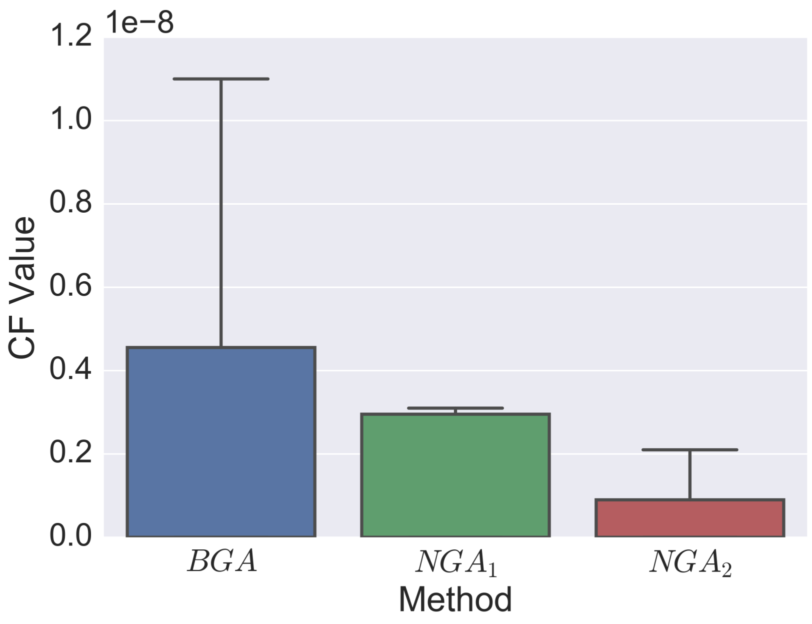


Figure 4.8: Cost function value after converged Nelder-Mead algorithm (median is at zero).

Figure 4.8 shows that no matter the method used to generate the initial guesses, the cost function values obtained after the Nelder-Mead algorithm are very low. However when starting from the estimates of NGA, the Nelder-Mead algorithm ends more often the search with lower cost function values. In figure 4.9a the estimated parameters for cost function values lower than 10^{-5} is presented.

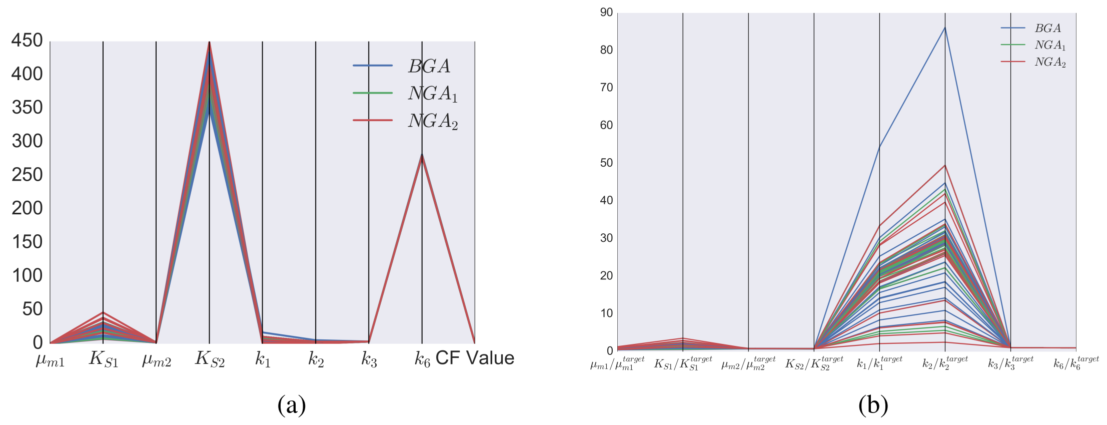
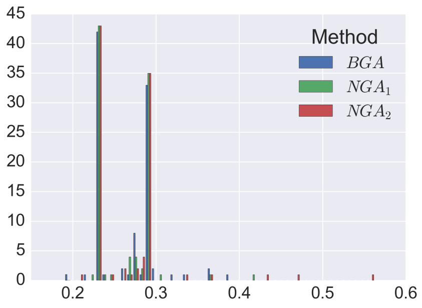


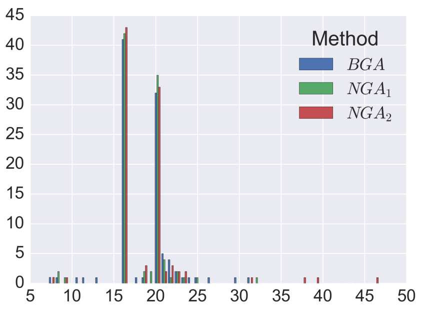
Figure 4.9: Parallel coordinates plot of all parameters with cost function values less than 10^{-5}.

Figure 4.9a suggests that K_{S1}, K_{S2} and k_1 are the parameters with greater dispersion however, dividing parameters by their target (figure 4.9b) k_1 and k_2 stands out that those are the worst ones followed by the K_{S1}. Moreover it seems to exist a relation between k_1 and k_2 since as k_1 grows k_2 grow as well with higher rate.

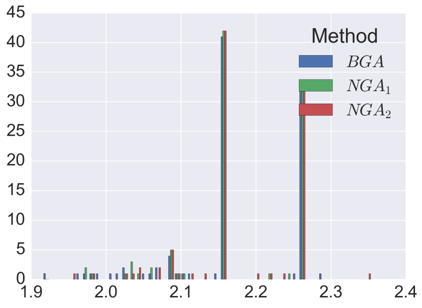
For better visualization of the individual parameter distributions, figure 4.10 shows the histograms of parameters grouped by the method that has produced the initial guess for the Nelder-Mead optimization and filtered for cost function values lower than 10^{-5}.



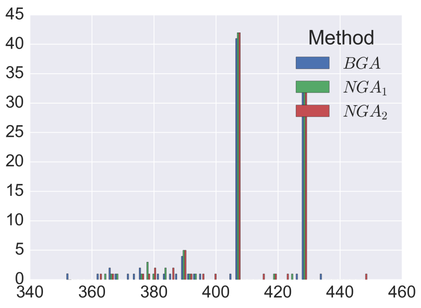
(a) $\mu_{m1}^{\text{target}} = 0.42912$



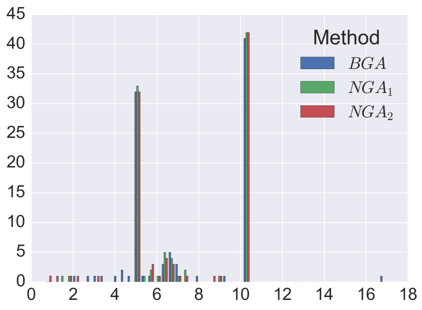
(b) $K_{S1}^{\text{target}} = 13.065$



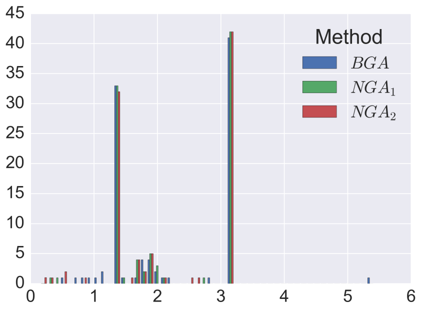
(c) $\mu_{m2}^{\text{target}} = 2.6493$



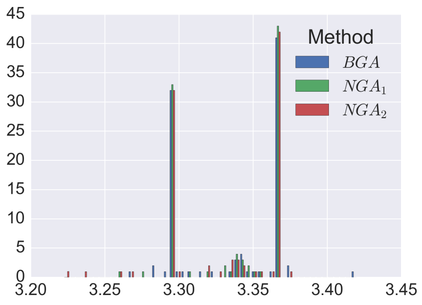
(d) $K_{S2}^{\text{target}} = 517.27$



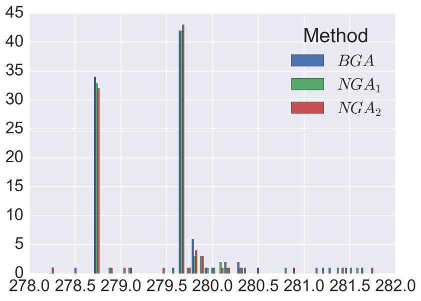
(e) $k_1^{\text{target}} = 0.31204$



(f) $k_2^{\text{target}} = 0.062776$



(g) $k_3^{\text{target}} = 3.1473$



(h) $k_6^{\text{target}} = 278.62$

Figure 4.10: Histogram of estimated parameters grouped by the method that has produced the initial guess.

What can be concluded from the previous figures is that, due to the low sensitivity, even for low tolerances, there is dispersion in estimated parameters. Moreover, filtering data for cost function values lower then 10^{-5}, two minima can be found. With this kind of results, it is impossible for the calibration procedure, at least from the practical point of view, to reach the correct set of parameters, as would be at first expected when using simulated results.

Furthermore, the statistical cost function values of the three GA methods used and filtered using the previous tolerance are presented in table 4.2.

Table 4.2: Last iteration of the Nelder-Mead algorithm from the 300 optimizations filtered for cost function values less than 10^{-5}.

count	285
mean	3.396867×10^{-07}
median	0.0×10^{00}
std	1.308811×10^{-06}
min	0.0×10^{00}
max	8.309300×10^{-06}
counts of cost function with value 0	220

Almost all runs performed has achieved the tolerance of 10^{-5}, moreover 220 of those have value 0. For those 220 runs with zero cost function value, figure 4.11 shows the methane evolution.

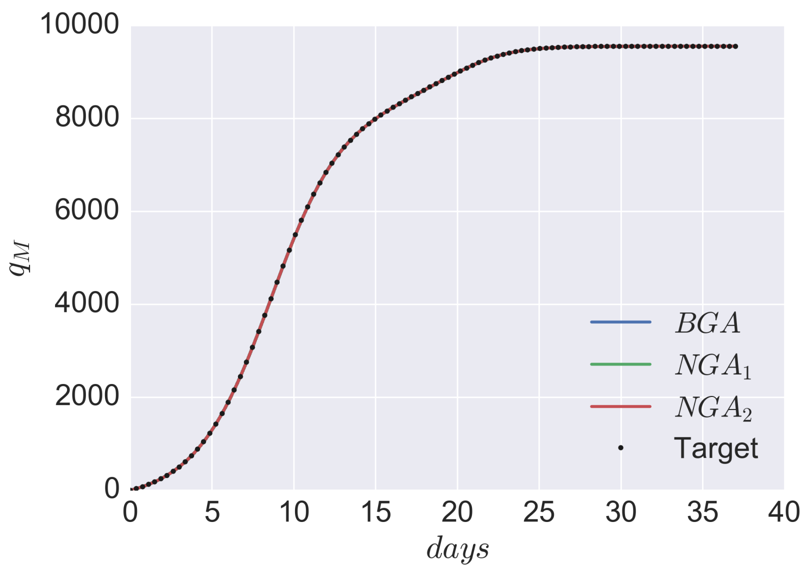


Figure 4.11: Methane obtained from the Nelder-Mead optimization for zero cost function values.

Despite the maladjusted parameters, the used variable for the cost function is well suited to the target. For the case of state variables, figure 4.11 shows that this adjustment is not as well estimated as the q_M variable.

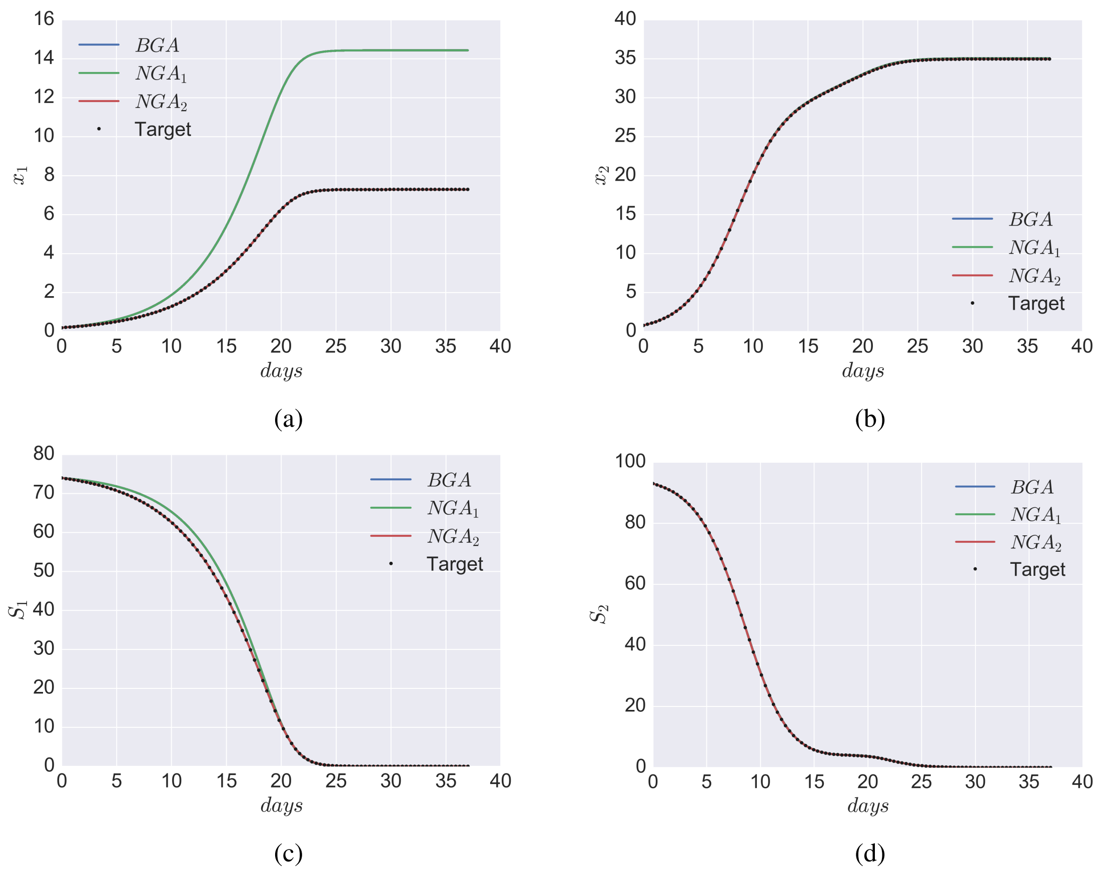


Figure 4.12: State variables obtained from the Nelder-Mead optimization for zero cost function values.

In particular, x_1 and S_1 has a misfitted case that is in accordance with the parameters histograms.

At the end, almost all runs were unable to approximate the parameter estimation, and none was able to estimate the exact value as would be expected. This is probably occurring due to the low sensitivity identified along the analysis. One way of overcome this problem may be achieved with the use of a different cost function.

Nevertheless, this proves that the direct use of this optimizations in real data without the understanding of the problem may produce misleading results.

Chapter 5

Conclusions

The first major conclusion of this work is that inverse optimization problems (IOP) applied to anaerobic digestion is extremely hard to solve. The difficulties arrive from multiple sources that are extremely hard by themselves. Identifiability analysis is a singular problem that is also very hard to solve. In this work, different combinations of observable variables and number of parameters to be estimated with two different software (DAISY and GENSSI) were performed. Both software failed to give an answer for the purposes of this work when the only available output data was biogas produced and all parameters for a batch situation were unknown. Despite that, they were useful to understand that if some parameters were known the problem would be, for most parameters, only locally identifiable.
Another difficulty observed along the work was the development of a sensitivity analysis that would fit the problem. Several strategies to overcome this problem were developed. Among them are the plots of the variation of pairs of parameter with the third dimension being the cost function value.
From this analysis, it was shown that the structure of the cost function is of extreme importance. The selection of a good cost function is a case of study by itself, that was not possible to develop in this work. What was observed from the selected cost function was that, independently of the parameters range, the cost function produces extremely low values. This has proved to be a great challenge to the optimization procedures, because numerical resolution is lost and the use of gradient based methods shows loss of numerical precision (machine eps), ending the optimization process extremely soon. In future, deeper studies of the cost function should be performed in order to overcome these difficulties.
Genetic algorithms (GA) have shown to be a good strategy for this kind of problems, nevertheless some difficulties were detected. Among them are the population saturation after few iterations, convergence rate decreases in very early stages and the computational expense of the algorithm. Due to these identified difficulties, a new adaptation to the method was developed that overcame the traditional GA in several aspects. This method, called Neighboured Genetic Algorithm (NGA), has

a faster decrease of the cost function value and the ability to achieve lower cost function values. Per iteration, it is computationally more expensive than the Basic Genetic Algorithm (BGA). Nevertheless, it has the ability of achieving better results in less time (due to less iterations needed), proving to be a good alternative to the BGA method. Despite that, there may be made improvements to the algorithm, for example, if a skewed distribution was used for the neighbor generation, convergence might be faster. This skewness could be originated from a gradient, a random value or any other measure. Another improvement to the NGA methods may reside in the way subjects are selected for neighboring. Since the first few subjects are usually close to each other, the use of a sequential selection may not produce the wider search that is needed. For that reason, selection of spaced subjects may be of interest since, despite some individuals having a higher cost function value, they may be near the optimum. However, if the optimum is in a deep valley, GA may need to be extremely lucky to select the right parameter combination that falls in such place. With NGA making use of the random distribution around this subject, the odds of falling in become better.

Despite the fact that GA is computationally expensive, its architecture is quite suitable for parallelization. Thus, the use of Python toolboxes in code development proved very useful since it was necessary to perform the optimization process 100 times to obtain statistical data for the comparison of the methods.

Despite all attempts, results after the GA optimization have remained considerably unacceptable. Therefore, a Nelder-Mead optimization was performed, using as initial guess the values obtained from BGA and NGA. The Nelder-Mead method was chosen because it does not use gradients which has proved to be a difficulty to this cost function. Even with these method results were not able to converge to the target parameters, but, from this results it was possible to identify some of the problems inherent to the model. If the k_2 parameter was zero, solution would still be possible to achieve, but in terms of the model, this would reject the existence of the x_1 and S_1 variables. In future it would be useful to investigate other AD models that would not produce this kind of difficulties in the calibration of the model.

Moreover, the outlier analysis, and their exclusion, has shown to be very useful for visualization and result interpretation, since results from the GA and from Nelder-Mead optimizations had a wide and very dispersed domain. Despite the exclusion of the outliers, filtering using the first and third quartiles was needed to further reduce the dispersed data and get readable results for plotting and detect clusterings.

References

[1] “http://www.iea.org/statistics/statisticssearch/, world polution emissions.” (cit. in page 2)

[2] “http://www.worldometers.info/world-population/, world population density.” (cit. in page 2)

[3] M. Ferreira, I. P. Marques, and I. Malico, “Biogas in Portugal: Status and public policies in a european context,” *Energy Policy*, vol. 43, pp. 267–274, 2012. (cit. in page 2)

[4] M. Raboni and G. Urbini, “Production and use of biogas in europe: a survey of current status and perspectives,” *Ambiente & Água - An Interdisciplinary Journal of Applied Science*, vol. 9(2), pp. 191–202, 2014. (cit. nas pág. 3, 4)

[5] L. Campestrini, D. Eckhard, and R. Rui, “Identifiability analysis and prediction error identification of anaerobic batch bioreactors,” *Control Automomation and Electrical Systems*, vol. 25, pp. 438–447, April 2014. (cit. nas pág. 5, 22, 49, 65)

[6] A. Donoso-Bravo, J. Mailier, C. Martin, J. Rodríguez, and C. A. Aceves-Lara, “Model selection, identification and validation in anaerobic digestion: a review,” *Water Res*, vol. 45, pp. 5347–64, Nov 2011. (cit. nas pág. 9, 19, 27, 28, 33)

[7] M. M. Fonseca and J. A. Teixeira, *Reactores Biológicos - Fundamentos e Aplicações*. Lidel, May 2015. (cit. nas pág. 9, 10, 11, 14)

[8] H. C. Lim and H. S. Shin, *Fed-batch cultures: principles and applications of semi-batch bioreactors*. Cambridge series in chemical engineering. (cit. in page 10)

[9] G. S. Weedermann, Marion and G. Wolkowicz, “Mathematical model of anaerobic digestion in a chemostat: effects of syntrophy and inhibition,” *J Biol Dyn*, vol. 7, pp. 59–85, 2013. (cit. in page 12)

[10] A. Lise, J. Baeyens, J. Degrève, and R. Dewil, “Principles and potential of the anaerobic digestion of waste-activated sludge,” *Elsevier*, 2008. (cit. nas pág. 12, 19)

[11] C. N. Hinshelwood, *The chemical kinetics of the bacterial cell.* Oxford: Clarendon Press, 1946. (cit. in page 15)

[12] A. R. Allman, *Fermentation Microbiology and Biotechnology.* CRC Press - Taylor and Francis Group, 2011. (cit. nas pág. 16, 17, 18)

[13] T. G. Müller, N. Noykova, M. Gyllenberg, and J. Timmer, "Parameter identification in dynamical models of anaerobic waste water treatment," *Math Biosci*, vol. 177-178, pp. 147–60, 2002. (cit. in page 20)

[14] N. Noykova, T. G. Müller, M. Gyllenberg, and J. Timmer, "Quantitative analyses of anaerobic wastewater treatment processes: identifiability and parameter estimation," *Biotechnol Bioeng*, vol. 78, pp. 89–103, Apr 2002. (cit. in page 20)

[15] L. Y. Lokshina, V. A. Vavilin, R. H. Kettunen, J. A. Rintala, C. Holliger, and A. N. Nozhevnikova, "Evaluation of kinetic coefficients using integrated monod and haldane models for low-temperature acetoclastic methanogenesis," *Water Res*, vol. 35, pp. 2913–22, Aug 2001. (cit. in page 20)

[16] L. Yu, P. C. Wensel, and J. Ma, "Mathematical modeling in anaerobic digestion (ad)," *Bioremed Biodeg*, 2013. (cit. in page 20)

[17] R. Antonelli, J. Harmand, J.-P. Steyer, and A. Astolfi, "Set-point regulation of an anaerobic digestion process with bounded output feedback," *IEEE*, 2003. (cit. in page 21)

[18] "Python ode solver using fortran," June 2001. (cit. nas pág. 22, 23)

[19] "Python ode solver using fortran." (cit. in page 23)

[20] A. Tarantola, *Inverse problem theory and methods for model parameter estimation.* Philadelphia, PA: Society for Industrial and Applied Mathematics, 2005. (cit. in page 28)

[21] L. Ming-Hua, T. Jung-Fa, and Y. Chian-Son, "A review of deterministic optimization methods in engineering and management," *Hindawi Publishing Corporation*, 2012. (cit. in page 29)

[22] R. Martí and G. Reinelt, *The linear ordering problem: exact and heuristic methods in combinatorial optimization*, vol. v. 175 of *Applied mathematical sciences*. Heidelberg: Springer, 2011. (cit. in page 30)

[23] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning.* Reading, Mass.: Addison-Wesley Pub. Co., 1989. (cit. in page 31)

[24] M. Tawarmalani and N. V. Sahinidis, *Convexification and global optimization in continuous and mixed-integer nonlinear programming: theory, algorithms, software, and applications*, vol. v. 65. Dordrecht: Kluwer Academic Publishers, 2002. (cit. nas pág. 31, 32)

[25] C. Grosan, A. Abraham, and H. Ishibuchi, *Hybrid evolutionary algorithms*. Studies in computational intelligence ; 75, Berlin: Springer, 2007. (cit. nas pág. 39, 40, 45, 46, 47)

[26] A. I. and S. N., "A review of genetic algorithm optimization: Operations and applications to water pipeline systems," *International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering*, vol. 7, no. 12, pp. 1262–1268, 2013. (cit. nas pág. 41, 42, 43)

[27] J. M. Mendes, "A comparative study of crossover operators for genetic algorithms to solve the job shop scheduling problem," *WSEAS TRANSACTIONS on COMPUTERS*, vol. 12, no. 4, pp. 164–173, 2013. (cit. nas pág. 43, 44)

[28] L. Wei and M. Zhao, "A niche hybrid genetic algorithm for global optimization of continuous multimodal functions," *Applied Mathematics and Computation*, vol. 160, no. 3, pp. 649–661, 2005. (cit. in page 44)

[29] J. H. Holland, "Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.," *MIT press*, 1992. (cit. in page 44)

[30] L. J. Eshelman, "The chc adaptive search algorithm: How to have safe search when engaging. foundations of genetic algorithms," 1991. (cit. in page 44)

[31] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 4, pp. 656–667, 1994. (cit. in page 44)

[32] Q. Yuan, Z. He, and H. Leng., "A hybrid genetic algorithm for a class of global optimization problems with box constraints," *Applied Mathematics and Computation*, vol. 197, no. 2, pp. 924–929, 2008. (cit. in page 45)

[33] J. Wang, O. K. Ersoy, M. He, and F. Wang, "Multi-offspring genetic algorithm and its application to the traveling salesman problem," *Applied Soft Computing*, vol. 43, pp. 415–423, 2016. (cit. in page 45)

[34] N. Krasnogor and J. Smith, “A tutorial for competent memetic algorithms: model, taxonomy, and design issues,” *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 5, pp. 474–488, 2005. (cit. in page 45)

[35] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby, “Hybrid genetic algorithms: A review.,” *Engineering Letters*, vol. 13, no. 2, pp. 124–137, 2006. (cit. in page 45)

[36] W. E. Hart, *Adaptive global optimization with local search*. PhD thesis, University of California, San Diego, 1994. (cit. in page 45)

[37] I. Ciornei and E. Kyriakides, “Hybrid ant colony-genetic algorithm (gaapi) for global continuous optimization,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, no. 1, pp. 234–245, 2012. (cit. in page 45)

[38] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby, “Hybrid genetic algorithms: A review,” vol. 13, no. 2, pp. 124–137, 2006. (cit. in page 45)

[39] A. A. Goldstein and J. F. Price, “On descent from local minima,” *Mathematics and Comptutaion*, vol. 4, no. 2, pp. 150–194, 1971. (cit. in page 50)

[40] J. Maomin and Y. Xin-She, “A literature survey of benchmark functions for global optimization problems,” *Inst. Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 2, pp. 150–194, 2013. (cit. in page 56)

[41] S. Cavaleiro Costa, I. Malico, F. M. Janeiro, A. P. Baptista, A. Coelho, T. Lopes da Silva, and I. P. Marques, “Dynamic modeling of biogas production supported by an experimental anaerobic digestion process.,” *11th Conference on Sustainable Development of Energy, Water and Environmental Systems*, 2016. (cit. in page 65)

[42] S. Cavaleiro Costa, F. M. Janeiro, and I. Malico, “On the use of genetic algorithms for parameter identification in anaerobic digestion modelling,” *12th International Conference on Diffusion in Solids and Liquids: Mass Transfer, Heat Transfer and Microstructures and Properties*, 2016. (cit. in page 65)