MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21c/2019/05.15.23.22-TDI

A NEW ADAPTIVE EVOLUTIONARY ALGORITHM FOR DESIGN OPTIMIZATION

Eric Demetrius de Castro Barroca

Master's Dissertation of the Graduate Course in Engineering and Space Technology/Space Systems Engineering and Management, guided by Drs. Fabiano Luís de Sousa, Ronan Arraes Jardim Chagas, and Fernando Manuel Ramos, approved in May 20, 2019.

URL of the original document:
<http://urlib.net/8JMKD3MGP3W34R/3TAJS9H>

INPE
São José dos Campos
2019

PUBLISHED BY:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GBDIR)
Serviço de Informação e Documentação (SESID)
CEP 12.227-010
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/7348
E-mail: pubtc@inpe.br

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21c/2019/05.15.23.22-TDI

A NEW ADAPTIVE EVOLUTIONARY ALGORITHM FOR DESIGN OPTIMIZATION

Eric Demetrius de Castro Barroca

Master's Dissertation of the Graduate Course in Engineering and Space Technology/Space Systems Engineering and Management, guided by Drs. Fabiano Luís de Sousa, Ronan Arraes Jardim Chagas, and Fernando Manuel Ramos, approved in May 20, 2019.

URL of the original document:
<http://urlib.net/8JMKD3MGP3W34R/3TAJS9H>

INPE
São José dos Campos
2019

Cataloging in Publication Data

Barroca, Eric Demetrius de Castro.

B278n A new adaptive evolutionary algorithm for design optimization / Eric Demetrius de Castro Barroca. – São José dos Campos : INPE, 2019.

xviii + 83 p. ; (sid.inpe.br/mtc-m21c/2019/05.15.23.22-TDI)

Dissertation (Master in Engineering and Space Technology/Space Systems Engineering and Management) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2019.

Guiding : Drs. Fabiano Luís de Sousa, Ronan Arraes Jardim Chagas, and Fernando Manuel Ramos.

1. Generalized extremal optimization. 2. Adaptive evolutionary algorithms. 3. Design optimization. 4. Space engineering. 5. Multidisciplinary optimization. I.Title.

CDU 629.78:005.591.1

Aluno (a): ***Eric Demetrius de Castro Barroca***

Título: **"A NEW ADAPTIVE EVOLUTIONARY ALGORITHM FOR DESIGN OPTIMIZATION"**

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de ***Mestre*** em

Engenharia e Tecnologia Espaciais/Eng. Gerenc. de Sistemas Espaciais

Dr. **Walter Abrahão dos Santos**

Presidente / INPE / São José dos Campos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dr. **Fabiano Luis de Sousa**

Orientador(a) / INPE / SJCampos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dr. **Ronan Arraes Jardim Chagas**

Orientador(a) / INPE / São José dos Campos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dr. **Fernando Manuel Ramos**

Orientador(a) / INPE / SJCampos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Este trabalho foi aprovado por:

( ) maioria simples

(X) unanimidade

São José dos Campos, 20 de maio de 2019

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de ***Mestre*** em

Engenharia e Tecnologia Espaciais/Eng. Gerenc. de Sistemas Espaciais

Dr. Roberto Luiz Galski

Membro da Banca / INPE / São José dos Campos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dr. Antônio Augusto Chaves

Convidado(a) / UNIFESP / São José dos Campos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Este trabalho foi aprovado por:

( ) maioria simples

(X) unanimidade

São José dos Campos, 20 de maio de 2019

“É necessário sempre acreditar que o sonho é possível, que o céu é o limite e você, "truta", é imbatível”.

Racionais
em *“A Vida é um Desafio”*, 2002

ACKNOWLEDGEMENTS

E. D. C. Barroca acknowledges the support given by the Coordination for the Improvement of Higher Education Personnel (CAPES).

ABSTRACT

In this work a new adaptive evolutionary algorithm derived from a stochastic algorithm for design optimization called Generalized Extremal Optimization (GEO) is introduced. It eliminates the single free parameter of GEO by controlling its value during the search by an adaptive approach which improved GEO performance significantly, even when considering the “best” GEO configurations. Nonetheless, it maintains the algorithm principal characteristics of dealing with continuous, discrete and integer design variables on convex or disjoint spaces while respecting design constrains. This new algorithm, called Adaptive Generalized Extremal Optimization (A-GEO), is implemented in two variations and applied to a multidisciplinary optimization problem of spacecraft engineering, showing the potential of the new methods in solving real engineering problems.

Keywords: Generalized Extremal Optimization. Adaptive Evolutionary Algorithms. Design Optimization. Space Engineering. Multidisciplinary Optimization.

UM NOVO ALGORÍTMO EVOLUTÍVO ADAPTIVO PARA OTMIZAÇÃO DE PROJETOS.

RESUMO

Neste trabalho um novo algoritmo evolutivo adaptativo derivado de um algoritmo estocástico para otimização de projetos chamado Generalized Extremal Optimization (GEO) é introduzido. Este elimina o único parâmetro livre presente no GEO através de um método adaptativo que controla os valores deste durante a busca, assim melhorando a performance do GEO significantemente, até mesmo quando comparada a sua “melhor” configuração. Porém, mantém suas principais características de lidar com variáveis de projeto continuas, discretas e inteiras em espaços convexos ou disjuntos respeitando as restrições de projeto. Este novo algoritmo, chamado Adaptive Generalized Extremal Optimization (A-GEO), é implementado em duas variações e aplicado a um problema de otimização multidisciplinar de engenharia de satélites, mostrando o potencial dos novos métodos em resolver problemas reais de engenharia.

Palavras-chave: Otimização Extrema Generalizada. Algoritmos evolutivos adaptativos. Otimização de projetos. Engenharia espacial. Otimização multidisciplinar.

LIST OF FIGURES

Page

1.1 Evolutionary algorithms workflow. 2
1.2 Hierarchy of parameter tuning. 4
1.3 Possible combinations of tuning and control. 4
1.4 Number of papers per year that discuss controlling each EA parameter. . 6
1.5 A conceptual model of adaptive parameter control in EAs. 7

2.1 GEO algorithm. 11
2.2 Design variables $X_n, n = \{1 \dots N\}$, are encoded in a binary string of L bits. 13

3.1 A-GEO Algorithms. 16
3.2 Papers classified based on the adaptive parameter control mechanisms. . 20

4.1 Design spaces shown in two dimensions for all test functions. 24
4.2 Overall performance F1 results for GEO and A-GEO. 30
4.3 F1 function evolution of τ parameter through generations. 31
4.4 Overall performance F2 results for GEO and A-GEO. 32
4.5 Rosenbrock function evolution of τ for A-GEO$_1$ and A-GEO$_2$. 33
4.6 Overall performance F3 results for GEO and A-GEO. 35
4.7 Griewangk function evolution of τ for A-GEO$_1$ and A-GEO$_2$. 36
4.8 Overall performance F4 results for GEO and A-GEO. 37
4.9 Rastringin function evolution of τ for A-GEO$_1$ and A-GEO$_2$. 38
4.10 Overall performance F5 results for GEO and A-GEO. 40
4.11 Schwefel function evolution of τ for A-GEO$_1$ and A-GEO$_2$. 41

5.1 Geometries of the orbit and the field of view of the satellite. 57
5.2 Satellite's cross-sectional area and dry mass relation. 60
5.3 DAS output vs Model. 62
5.4 XDSM for spacecraft conceptual design. 67
5.5 Spacecraft conceptual design search space. 70
5.6 Spacecraft conceptual design GEO Tunning. 71

LIST OF TABLES

Page

3.1 A-GEO classification on Aleti and Moser (2016) parameter control framework. 19

4.1 Test functions for performance experiment. 23
4.2 GEO, A-GEO$_1$ and A-GEO$_2$ tunning versus controlling configurations. . 25
4.3 GEO tunning τ configurations. 26
4.4 Tunning versus controlling F1 results for GEO and A-GEO. 26
4.5 Tunning versus controlling F2 results for GEO and A-GEO. 26
4.6 Tunning versus controlling F3 results for GEO and A-GEO. 27
4.7 Tunning versus controlling F5 results for GEO and A-GEO. 27
4.8 GEO, A-GEO$_1$ and A-GEO$_2$ overall performance general configurations. 29
4.9 F1 τ average results. 31
4.10 Rosenbrock τ average results. 34
4.11 Griewangk τ average results. 36
4.12 Rastringin τ average results. 39
4.13 Schwefel τ average results. 40
4.14 Summary of the CEC2017 Test Functions. 45
4.15 Error values of A-GEO$_2$ and TLBO-FL on the 29 benchmark functions for $D = 10$. 49
4.16 Error values of A-GEO$_2$ and TLBO-FL on the 29 benchmark functions for $D = 30$. 51

5.1 Simulation input settings. 68
5.2 Optimal solution. 69
5.3 Microsoft Surface 3 Pro Specifications. 69
5.4 Simulation results. 72

CONTENTS

Page

1 INTRODUCTION 1

1.1 Parameter setting on evolutionary algorithms 2

1.2 Parameter control techniques 5

2 GEO ALGORITHM 11

3 ADAPTIVE GENERALIZED EXTREMAL OPTIMIZATION (A-GEO) ALGORITHM 15

3.1 The A-GEO algorithm 15

3.2 A-GEO parameter control mechanism classification 18

4 PERFORMANCE EXPERIMENTS AND ANALYSIS 23

4.1 Tunning versus controlling mechanism 25

4.2 A-GEO performance experiments and τ analysis 28

4.2.1 Test Function 1 (DeJong #1) 29

4.2.2 Test Function 2 (Rosenbrock Function) 31

4.2.3 Test Function 3 (Griewangk Function) 34

4.2.4 Test Function 4 (Rastringin Function) 37

4.2.5 Test Function 5 (Schwefel Function) 39

4.3 A-GEO: A Glance Into the Future 42

4.3.1 2017 IEEE Congress of Evolutionary Computation Benchmark 43

4.3.2 Teaching Learning Based optimization with Focused Learning (TLBO-FL) 44

4.3.3 Statistical Benchmark Comparison 47

5 SPACECRAFT CONCEPT DESIGN APPLICATION 55

5.1 Orbit analysis model 55

5.1.1 Nominal orbit 56

5.1.1.1 Field of View 57

5.1.2 Orbit acquisition 58

5.1.3 Orbit maintenance 59

5.1.4 Deorbit 61

5.2 Payload model 63

5.3 Propulsion model 64
5.4 Satellite model 64
5.5 Multidisciplinary Optimization Framework 66
5.6 Spacecraft concept design MDO Results 68

6 CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH 73

REFERENCES 77

1 INTRODUCTION

The concept design of satellites and its optimization has been real challenging to engineers, specially in early phases of development (CHAGAS et al., a; FLIEGE et al., 2012; WERTZ et al., 2011). However, use satellite optimization techniques in the very first phase of the project, also called pre-phase A or conceptual design phase, is not an impossible task. As showed by Chagas et al. (b) and Chagas et al. (a), in these cases, there is very little information about the mission and the design team must find feasible solutions based solely on the high-level mission specification provided by the stakeholders. Hence, as stated in Chagas et al. (b), Multidisciplinary Optimization (MDO)[1] techniques can be used to search the design space for optimal solutions, giving the design team a starting point for the analysis.

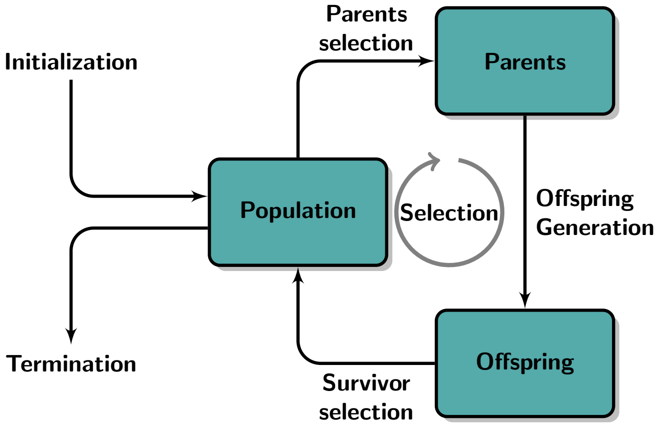
MDOs integrate models of diverse disciplines (*e.g.* orbital analysis, structural, electrical, etc.) and optimize the design variables chosen to achieve objectives, which are given as mathematical functions, through numerical computation (MARTINS; LAMBE, 2013). The optimization cycle of an MDO can be applied by diverse algorithms presented in the literature, from local search algorithms to neural networks. One of these possibilities is to use evolutionary algorithms. They have been applied to diverse science and engineering problems through the years and found to be a suitable tool to be added to the engineer's toolbox (EIBEN; SMITH, 2015; GOSSELIN et al., 2009; LIAN et al., 2010; SCHMIDT; LIPSON, 2009; SHIRAZI, 2015).

In fact, evolutionary algorithms are applicable to almost any kind of optimization problem. The concept of evolutionary algorithms comes from the theory of natural selection and survival of the fittest presented in Darwin (1859). Following this principle, the first algorithms were developed to simulate the natural selection behavior focusing on help biologists and naturalists have a better understanding of the evolution of species, until 1970. After it, Evolutionary Strategies, Evolutionary Programming (EIBEN; SMITH, 2003) and Genetic Algorithms (GAs) (HOLLAND, 1975) emerged. GAs where introduced by Holland (1975) and later popularized by Goldberg (1989). After GAs researchers started to incorporate other natural features or behaviors into their optimization algorithms, as in Particle Swarm Optimization (PSO) (KENNEDY; EBERHART, 1995), Ant Colony Optimization (ACO) (DORIGO et al., 1999) and Generalized Extremal Optimization (GEO) (SOUSA, 2003; SOUSA; RAMOS, 2003). Thus, optimization methods influenced by the laws of nature started to become really popular, being them evolutionary or not.

[1] For more information about MDO and its architectures see Martins and Lambe (2013).

Paying particular attention to evolutionary algorithms, one can say that they consist of five main characteristic: a population of individuals, a representation for the individuals, a notion of fitness, a life and death cycle ruled by the adaptability (fitness), called selection, and a notion of heredity. Those characteristics put together, as in Figure 1.1, represent a complete evolutionary algorithm.

Figure 1.1 - Evolutionary algorithms workflow.



SOURCE: Made by the author.

The main reason for evolutionary algorithms popularity is their universal applicability. They can be used to solve a vast number of problems once their free parameters values, such as population size and mutation rate, are adjusted and the problem is encoded in the form of a population of individuals. However this takes a toll in their performance. Since they are not a problem specific application they may overlook important aspects of the problem search space and progress slower than an specific heuristic.

1.1 Parameter setting on evolutionary algorithms

Although EAs performance on a given application be a function of the problem being addressed and the kind of EA being used, another common aspect that has great influence over it is the proper setting of their free parameters (KARAFOTIAS et al., 2015).

One technique used for parameter setting, besides intuition or convention, is called tuning. Tuning is a process that calibrates the EA to better adjust it to the problem being solved, since the EA parameters setting vary from problem to problem. Therefore, each time that a new problem is to be solved through evolutionary computation the algorithms need to be tuned again (ALETI; MOSER, 2016).

Tune an algorithm can be a very time consuming process that asks of the user a good understanding of the algorithm, in order to extract its best performance (ALETI; MOSER, 2016; CERVANTES; STEPHENS, 2009; EIBEN et al., 1999). Also, in problems where the objective function or the design space have a complex landscape the best algorithm performance can not be obtained by a static parameter configuration, in other words, there is no static optimal configuration (CERVANTES; STEPHENS, 2009; GOLDMAN; TAURITZ, 2011; SMITH; FOGARTY, 1996).

Fortunately, with an oversimplification we could say that: "[...] the tuning problem has been solved by now. At least, there are very good parameter tuning methods developed and publicized over the last decade and the EC community is increasingly adopting them." (KARAFOTIAS et al., 2015, p. 1). [2]

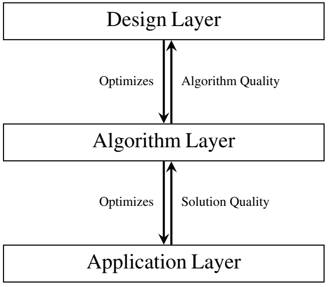
Although parameter tuning being a extremely explored approach, there exists another option to set parameters values in EAs. This technique is called parameter control (ALETI; MOSER, 2016; KARAFOTIAS et al., 2015). It can set the EA parameters values during the search and provides some advantages (KARAFOTIAS et al., 2015):

- Possible use of appropriate parameter values in different stages of the search.
- When facing dynamic problems, allows the EA to adjust to the changing fitness landscapes.
- Can collect information about the fitness landscape during the search and use it to improve the algorithm performance.
- Frees the user from choosing parameter values, solving the tuning problem.

Now that both strategies have been presented, a general framework developed in Eiben and Smit (2011) for parameter setting is shown in Figure 1.2. This framework helps to understand how a parameter setting strategy (*e.g.* parameter tuning and parameter control) can act upon an EA and what differentiates one from the other.

[2] For more information, an overview on tuning mechanisms subject was made in Eiben and Smit (2011).

Figure 1.2 - Hierarchy of parameter tuning.



Control flow (left arrows) and information flow (right arrows) through the three layers in the hierarchy of parameter tuning.

SOURCE: Adapted by the author from Eiben and Smit (2011) and Karafotias et al. (2015).

As pictured in Figure 1.2 there are three layers in this framework. According to Karafotias et al. (2015, p. 3) they are: "[...] the application layer (that contains a problem to be solved), the algorithm layer (that contains an EA), and the design layer (that contains a method to specify all details of the given EA, that is, its numeric as well as symbolic parameters).". Also, this scheme can be divided into two optimization problems. The first is the EA trying to find an optimal solution for the current problem and is inside the Application + Algorithm layers (KARAFOTIAS et al., 2015). The second problem belongs to the Design + Algorithm layers, and consists of trying to find the optimal parameter settings for the EA (KARAFOTIAS et al., 2015).

Thus, this framework presents two possibilities on where to place the parameter setting strategy depending on their behavior: The design or algorithm layer (KARAFOTIAS et al., 2015). Figure 1.3 shows how the parameter setting strategy can be characterized between control and tuning based on the layer chosen for its placement.

Figure 1.3 - Possible combinations of tuning and control.

		Control	
		Yes	No
Tuning	Yes	Control mechanism tailored to application	Static values obtained with tuning
	No	Control mechanism used out-of-the-box	Static values based on intuition or convention

SOURCE: Adapted by the author from Karafotias et al. (2015).

Placing the parameter setting on the design layer gives two possible outcomes: a tuning strategy is used offline and the EA parameters will remain static during the search or a control technique is used and they will change during the search in a online fashion way (KARAFOTIAS et al., 2015). This means that the parameter setting algorithm can be separated from the EA algorithm and used independent of it. However, if the parameter setting strategy is placed on the algorithm layer the EA and parameter strategy become one and only (KARAFOTIAS et al., 2015). This implies that these control parameter techniques have parameters of their own to be tuned, although they are hidden behind design decisions (KARAFOTIAS et al., 2015). Therefore, a control parameter technique could be tuned.

1.2 Parameter control techniques

Algorithms that are able to set their own parameters started to become popular in the EA community after 1990. Not that the parameter control problem did not existed before, but it passed unnoticed through the years (KARAFOTIAS et al., 2015). One reason that contributed to this was a misleading or missing nomenclature unification. Situation that changed with Eiben et al. (1999) publication that presented a clear and unifying classification for parameter control techniques (KARAFOTIAS et al., 2015).

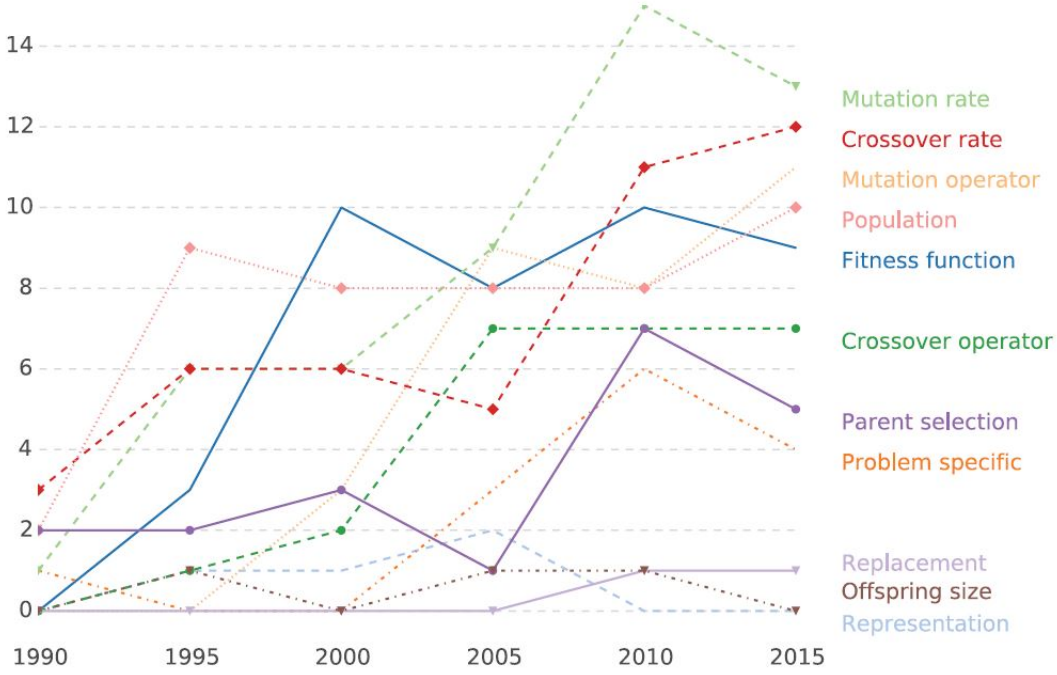
In Eiben et al. (1999) work, parameter control techniques were classified as: deterministic, adaptive and self-adaptive strategies (EIBEN et al., 1999; KARAFOTIAS et al., 2015). **Deterministic** strategies change the value of the algorithm free parameters based on a given schedule (*e.g.* as function of the number of iterations)(EIBEN et al., 1999; KARAFOTIAS et al., 2015). **Adaptive** strategies (ALETI et al., ; GINLEY et al., 2011; VAFAEE; NELSON, 2010) change the algorithm parameters values based on a given heuristic formula that uses feedback of the population (EIBEN et al., 1999; KARAFOTIAS et al., 2015) and in **self-adaptive** versions, they encode the algorithm free parameters in the population, such that they are co-evolved with it along the search (ALETI; MOSER, 2016; CERVANTES; STEPHENS, 2009; EIBEN et al., 1999; JITKONGCHUEN; THAMMANO, 2014; KARAFOTIAS et al., 2015).

Furthermore, control mechanisms can also be classified in respect to their designed purpose. If a control mechanism was designed to be used only for one specific parameter (*e.g.* Ahrari and Shariat-Panahi (2015), Lobo and Lima (2007)), it is called **parameter specific** (KARAFOTIAS et al., 2015). Combinations of heterogeneous control strategies characterizes **ensembles** (*e.g.* Nadi and Khader (2011)), used for multiple parameter algorithms optimization (KARAFOTIAS et al., 2015). Finally,

parameter independent methods can be applied to control any parameter (*e.g.* Wong et al. (2003)), with some limitations (KARAFOTIAS et al., 2015).

All of these forms of control approaches are applied to different classes of algorithms, which have different types and number of parameters to be adapted (ALETI; MOSER, 2016; KARAFOTIAS et al., 2015). A more recent systematic review was made by Aleti and Moser (2016), where they mapped papers that discussed parameters controlled in EAs per year since 1990 until 2016, as shown in Figure 1.4. As can be seen the most popular parameters adapted are mutation and crossover rates, and population size. As for the least popular ones they are replacement, representation and offspring size.

Figure 1.4 - Number of papers per year that discuss controlling each EA parameter.



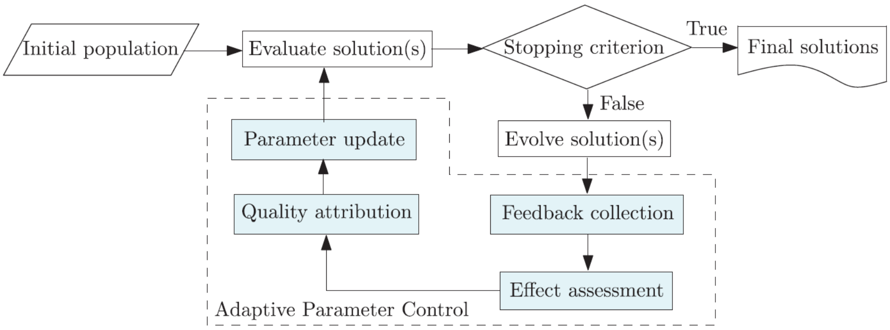
SOURCE: Adapted by the author from Aleti and Moser (2016)

In Aleti and Moser (2016, p. 6) parameter control is defined as a technique that: “[...] addresses the requirement of finding optimal parameter configurations as the search proceeds. It describes a process in which optimization starts with suboptimal parameter values that are adapted during the progress of the algorithm.”. This definition can be represented as a set V of parameters to be optimized:

$$\begin{aligned} & \underset{V}{\text{Optimize}}\ V = \{v_1, \ldots, v_n\} \\ & \text{Where } v_i \in \{v_{i1}, \ldots, v_{im}\} \\ & \text{Subjected to } m \leq m_u \end{aligned} \tag{1.1}$$

where n is the number of parameters and m_u is the upper limit for the number of possible values of the parameter. The goal of controlling a parameter is to find what combination of parameters values $V = \{v_ij, \ldots, v_kl\}$ optimizes the performance of the algorithm in a time t. Aleti and Moser (2016) revised most of the controlling mechanisms presented in published works in the evolutionary computing field and developed a concept model, shown in Figure 1.5, that shows step by step how a algorithm free parameter can be set using parameter control mechanisms during the search.

Figure 1.5 - A conceptual model of adaptive parameter control in EAs.



SOURCE: Adapted by the author from Aleti and Moser (2016)

The framework proposed by Aleti and Moser (2016), has four main steps that are performed to set a parameter through controlling mechanisms, shown as painted box in Figure 1.5. They are feedback collection, effect assessment, quality attribution and parameter update. The majority of controlling techniques follow these steps to optimize a free parameter through the search, even tough they are not required to implement all of the steps.

First of all, according to the proposed model, to control a parameter it is necessary to identify a property of an EA that can be measured to assess the effect this parameter is inducting on the algorithm. This task is performed by the feedback collection strategy and the properties selected can be divided into five categories according to their type (ALETI; MOSER, 2016). If the feedback collection property selected is a solution quality or the relative quality of a set of solutions the approach is called phenotype or relative phenotype feedback (ALETI; MOSER, 2016). Similar approaches to the two previous explained are genotype and genotype diversity feedback that choose components or parts of a solution or set of solutions (*e.g.* building blocks) as the properties for the feedback collection phase (ALETI; MOSER, 2016). The last type of feedback collection is classified by the violation of constraints, called feasibility feedback (ALETI; MOSER, 2016). It is noteworthy, that as mentioned previously, some parameter control methods do not use feedback collection (*e.g.* deterministic control) or use it in a implicit manner as in self-adaptive parameter control algorithms (ALETI; MOSER, 2016).

Finished the feedback collection stage the next step is to indeed find a reference point to measured the effect the free parameter(s) has on the EA taking in consideration the property chosen in the feedback collection. Thus, it is initiated the effect assessment stage, which measures the impact of a free parameter on a certain phase of the EA relative to other phase (ALETI; MOSER, 2016). Divided in Aleti and Moser (2016) into ancestor, population, best, worst, median and current effect categories it uses, mostly, statistic measurements to evaluate the effects of a parameter. Ancestor effect measures the improvement of a solution in respect to this solution parents and population effect in respect to the population itself (ALETI; MOSER, 2016). Best, worst, median and current effect assessment measure the improvement in reference to the best, worst, median and current solution (ALETI; MOSER, 2016).

Right now, it is already known the property which gives us feedback about the quality of a free parameter and how and in reference to what its effect is measured. Thus, the next logical step is to finally estimate the parameter quality, a stage called quality attribution in Aleti and Moser (2016). This stage is responsible to calculate the quality of a parameter already applied to the search to make a better judgment on the next parameter value to be chosen, using the effect assessment metrics already computed (ALETI; MOSER, 2016). There are four categories that can characterize a strategy for quality attribution: immediate, average, extreme and learned. If the quality attribution is made relative to properties of the current solution it is in the immediate category, else if it is made relative to change to properties of a set or subset of solutions it is in the average category (ALETI; MOSER, 2016). In cases where the quality attribution looks for outliers or best changes in properties it falls into the extreme category and finally if some machine learning or forecasting technique is used it belongs in the learned category (ALETI; MOSER, 2016).

Finally, the parameter has been already analyzed in respect to a chosen property of the EA, its effect measured and its quality estimated leading to the last stage, the parameter value update. Parameter update is the last step on parameter control methods, where the parameter value is finally updated to be used on the next iteration. According to Aleti and Moser (2016) there are four categories of updates: quality proportionate, quality proportionate with minimum probability, greedy and deterministic. Quality proportionate and quality proportionate with minimum probability use a probability vector associated to the quality attribution to update its parameters, the difference between them is that the second approach has a minimum value for the probabilities to avoid that a certain parameter value that isn't good for the search in the moment gets lost (ALETI; MOSER, 2016). Greedy updates selects the best parameter value always and deterministic approaches have predefined rule to update parameter values (ALETI; MOSER, 2016).

All these steps explained summarize the process of controlling a parameter through the search, which can be applied to update a parameter value or even to choose between parameters to be applied on the next iteration. Although using parameter control may add some complexity to the algorithm, as shown by the explanation made above, it can yield better results than only tuning them for a fixed set of values, even if the control in itself is not optimal (EIBEN et al., 1999). For example, it has been observed that finding a function p(t) that changes a given free parameter “s” over the algorithm iterations, in a way that the search is to a given extent tuned over time, generates better results than keeping the value of the parameter constant, even if p(t) function is “suboptimal" (EIBEN et al., 1999).

New proposed evolutionary optimization methods have a lot to benefit in incorporating some strategy of adaptation for its free parameters, not only for improving its capacity of finding the best solution, but also as a practical way to avoid the process of searching the proper set of parameter values that will result in algorithm good performance for a given application. This work proposes a new version of GEO, called Adaptive Generalized Extremal Optimization (A-GEO), that applies an adaptive control mechanism to the algorithm with the goals of:

- Elaborate an adaptive control mechanism for GEO that controls the τ parameter through the search, eliminating the tunning process.

- Identify the advantages and disadvantages of using variations of an adaptive control mechanism on GEO versus tuning the algorithm.
- Discover how A-GEO performs against new state of the art optimization algorithms on a well established benchmark.

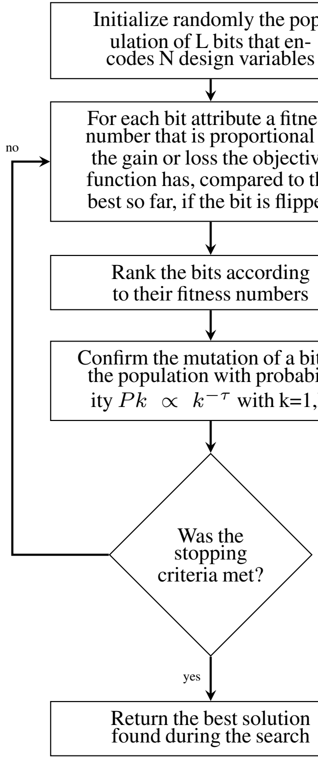
Finally, this dissertation is divided as follows: in Chapter 2 the GEO algorithm is presented, followed by the proposed algorithm, A-GEO, in Chapter 3. In Chapter 4 A-GEO performance and behavior are analyzed with a series of test functions experiments. In Section 4.3 A-GEO performance is compared with 2017 IEEE Congress of Evolutionary Computing, CEC2017 (AWAD et al., 2017) least performatic algorithm: Teaching Learning Based Optimization with Focused Learning, TLBO-FL (Kommadath; Kotecha, 2017). In Chapter 5 a spacecraft conceptual design optimization problem is presented with an MDO implementation using A-GEO as optimizer. Finally, in Chapter 6 the main conclusions of this work are presented alongside future work recommendations.

2 GEO ALGORITHM

GEO is a global search evolutionary meta-heuristic which has been successfully applied to many engineering optimization problems (ALBUQUERQUE et al., 2016; CUCO et al., 2009; FREITAS et al., 2018; MURAOKA et al., 2006; KUMAR et al., 2017; SWITALSKI; SEREDYNSKI, 2008; VLASSOV et al., 2006). Different implementations of GEO have been proposed since its conception (COELHO et al., 2017; GALSKI, 2007; GALSKI et al., 2009; GALSKI et al., ; MAINENTI-LOPES et al., 2012; LOPES et al., 2016; XIE et al., 2009), but strategies for the adaptation of its free parameters have been little explored.

The GEO algorithm, shown in Figure 2.1, is based on the Extreme Optimization (EO) method developed by Boettcher and Percus (2001). It was proposed as a way to generalize the algorithm so it could be applied to a broad class of optimization problems.

Figure 2.1 - GEO algorithm.


Initialize randomly the population of L bits that encodes N design variables
For each bit attribute a fitness number that is proportional to the gain or loss the objective function has, compared to the best so far, if the bit is flipped
Rank the bits according to their fitness numbers
Confirm the mutation of a bit of the population with probability $Pk \propto k^{-\tau}$ with k=1,L
Was the stopping criteria met?
no
yes
Return the best solution found during the search

SOURCE: Adapted by the author from Sousa and Ramos (2003).

The canonical GEO is composed of one free parameter, named τ, and a binary string population that encodes the design variables similar to genetic algorithm chromosomes (SGA) (GOLDBERG, 1989). Each bit of this string is a specie (SOUSA; RAMOS, 2003), as shown in Figure 2.2, where their precision p is defined by the number of bits m they are encoded respecting an upper and lower boundary X_n^u and X_n^l:

$$2^m \geq [(X_n^u - X_n^l)/p + 1] \tag{2.1}$$

The value of each variable is encoded into the string, and the objective function must be evaluated to compute the fitness of each specie, so the value of the variable n encoded in binary is converted to their decimal values by:

$$X_n = X_n^l + (X_n^u - X_n^l)[I_n/(2^m - 1)] \tag{2.2}$$

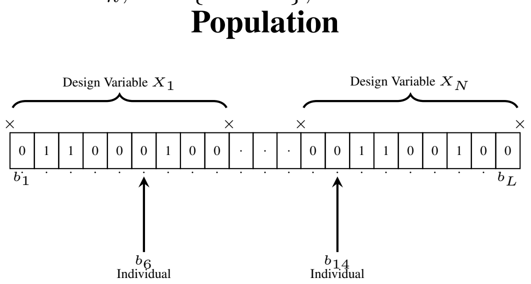
where I_n is the integer number obtained in the conversion from its binary value.

The fitness value, which is a way to measure how adapted this specie is in relation to the others, is attributed proportionally to the gain or loss in the value of the objective function resulted from a bit flip in the population. For example, in a minimization problem, if flipping bit b_6 in Figure 2.2 results in a higher value of objective function for the entire population than if flipping bit b_{14}, then bit b_6 is more adapted (fitter) than bit b_{14}. Note that in GEO firstly the flip is done in a bitwise process, only for the fitness attribution. That is, during the attribution of the fitness only the bit that is being assessed is flipped, all others remain with their current value. After the fitness is attributed for all bits they are ranked and one of them is mutated with probability proportional to its ranking. At each canonical GEO iteration (generation), only one bit is confirmed to mutate. In a slightly variation of it, called $\mathrm{GEO}_{\mathrm{var}}$, a bit is mutated for each variable. In either version, The process of fitness attribution, ranking and mutation is repeated until a given stopping criterion is met. The population of bits is evolved through this process and the best configuration found during the search is returned at the algorithm stop.

The τ parameter controls the selection pressure, making the algorithm more deterministic for higher values and more stochastic for lower ones. This gives the algorithm the capability to escape local optimums due to being able to walk through the space of the problem not necessarily improving the best objective function value found in some iteration (SOUSA; RAMOS, 2003). It has been extensively studied and in practice, it has been observed that the best τ for different applications lies in the range [0.75 to 5.0] (SOUSA; RAMOS, 2003; SOUSA et al., 2005; SILVA NETO et al., 2016;

SOUSA, 2003). Besides, the effect that τ has on the algorithm may indicate traces of Self Organized-Criticality (SOC) behavior in GEO (SOUSA, 2003).

Figure 2.2 - Design variables $X_n, n = \{1 \dots N\}$, are encoded in a binary string of L bits.



SOURCE: Made by the author.

SOC (BAK; CHEN, 1991) is a model created to describe dynamic systems behaviors and applied to explain the process of evolution. Basically a system stays at equilibrium for an amount of time until it reaches a critical point, which causes instability into the system. This system suffers "avalanches" that can be really small or even the size of the entire system, what makes the system return to its equilibrium point (BAK; CHEN, 1991). SOC may be present on GEO due to the τ parameter that improves the population through avalanches, however more studies are necessary to prove this behavior (SOUSA, 2003).

Along the previous years diverse GEO variations were developed, versions that explored methodologies as a auto reset for every time the algorithm stagnated, one for parallel computation, a multi-ranking to make it more symmetric and a multi-objective version are some examples (GALSKI, 2007).

Lopes (2013), Lopes et al. (2016) elaborated a base ten version of GEO, called GEO-real, and a multi-objective version of it, in both the design variables assumed their continuous values instead of being encoded in binary, and suffered mutation by a Normal distribution.

Considering concurrent populations approaches there are few works in the literature where EO or GEO are transformed into a multi population based algorithm (CHEN et al., 2006; ZENG et al., 2014; XIE et al., 2009) as far as the author knowledge goes. In Xie et al. (2009) GEO is extended into two versions that work with continuous variables.

First, population based GEO (PGEO) implements multiple populations that are ranked according to their fitness to then be selected by the same selection process of GEO, using τ. After, the selected individuals variables are mutated through the same selection process, but with a different τ, changing the individual for the next generation. The second version, named Hybrid GEO (HGEO), is an extension of the first, where its hybridized with GA. Basically, it runs PGEO, GA crossover and elitist operator for the selected populations.

Galski (2007) also developed a version of GEO that presented different possibilities for encoding the design variables. Introducing a new parameter b to represent the base in which the user would like to encode the problem variables. A deterministic hybrid version with Simulated Annealing (GALSKI, 2007; GALSKI et al., 2009), also developed by Galski, consisted in using the temperature schedule of the simulated annealing merged with the τ parameter of GEO. To define the schedule it is necessary to set the number of stages, the number of function evaluations and the value of τ in each stage. Although such approach allows the variation of τ during the search, in a deterministic way, it added to the algorithm two new free parameters.

All those versions never explored controlling the τ parameter, except for the hybrid GEO/SA one. Even this version presented a deterministic way to mutate τ through the search and yet added another parameters for the user to tune.

3 ADAPTIVE GENERALIZED EXTREMAL OPTIMIZATION (A-GEO) ALGORITHM

GEO having only one free parameter to be adapted makes it a good candidate for the use of control mechanisms from the implementation simplicity point of view, since there would be no interference of other parameters on its performance. Plus, in Aleti and Moser (2016), Karafotias et al. (2015) reviews, it has been shown that between the identified parameters controlled on published works since 1990, the parent selection receives little attention from the community. Thus, since there is little information on the scientific community about control mechanisms actuating on selection parameters it is interesting the use of control mechanisms within GEO.

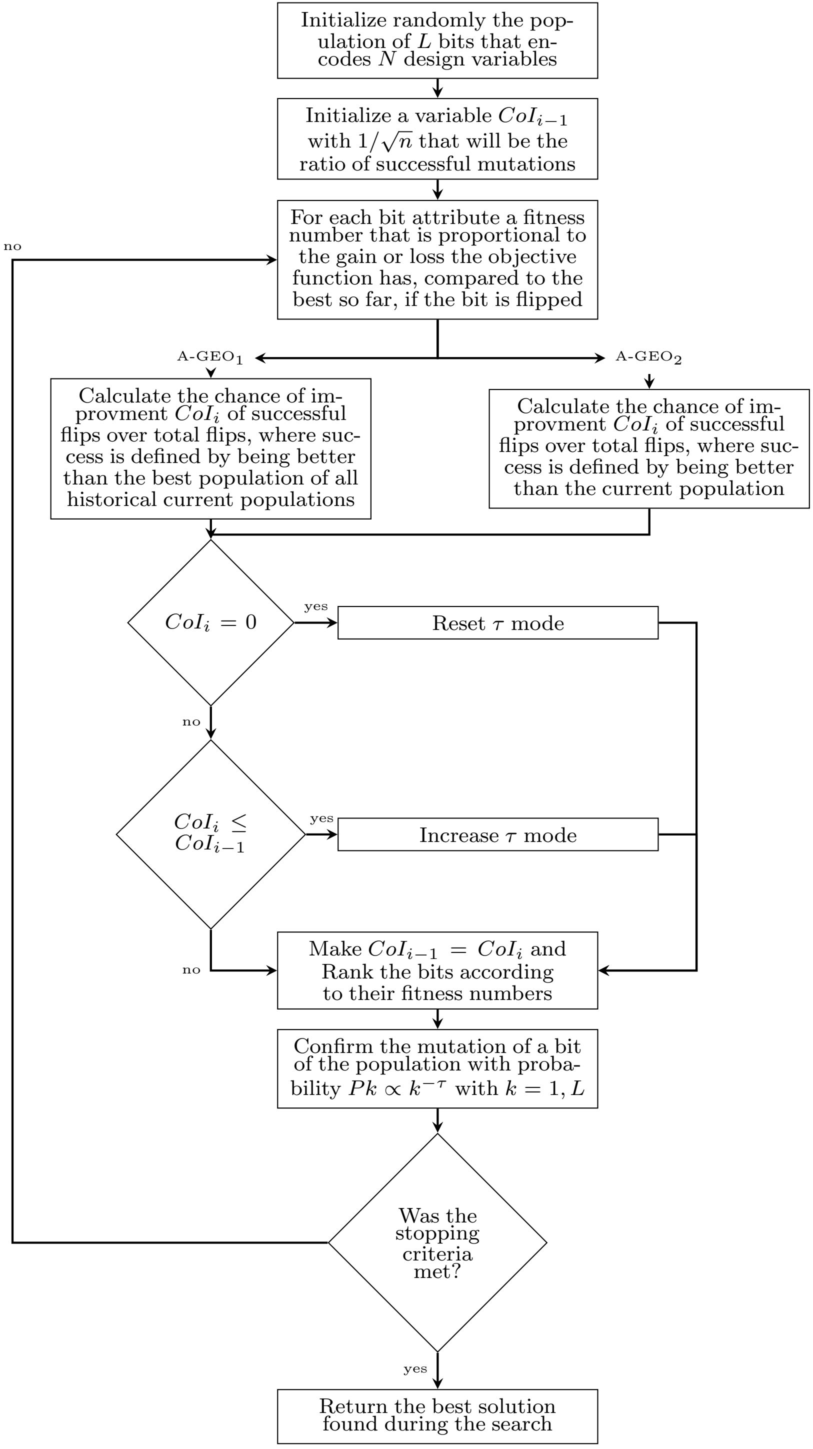
In the past Galski et al. (2009) developed the first version of GEO that employed parameter control techniques. This version updated τ value based on a schedule derived from Simulated Annealing (KIRKPATRICK et al., 1983), as described in Chapter 2. Unfortunately, it added two more parameters to GEO and had a fixed rate of change, which did not establish a proper balance between exploration and exploitation during the search. Next Galski et al. () developed a self-tuning version of GEO thorough an hybrid approach named GEO + ES. This version represented the design variables as real numbers, implemented a new mutation operation and eliminated the τ parameter. However, it still added three other adjusting parameters: the learning rate, the learning bias and the number of mutations.

Thus, with those two goals in mind, make GEO parameterless and balance exploration and exploitation along the search, A-GEO was developed. To achieve its goals A-GEO employs an adaptive parameter control technique to change τ values at each generation i.

3.1 The A-GEO algorithm

A-GEO was split into two GEO variations, A-GEO$_1$ and A-GEO$_2$, presented in Figure 3.1. To balance the exploitation and exploration of A-GEO the GEO capacity of exploration was maintained while the exploitation of local optimums was improved. This was achieved by oscillating between behaviors proportionated by different τ values. First, to increase exploitation of valleys, τ is made more deterministic when the current population has lesser or equal chances of improving the solution than the previous iteration population. Second, τ is made more stochastic when the current population has no more ways to improve the solution, giving the algorithm more randomness, enabling it to explore more points on the search space.

Figure 3.1 - A-GEO Algorithms.


Initialize randomly the population of L bits that encodes N design variables
Initialize a variable CoI_{i-1} with $1/\sqrt{n}$ that will be the ratio of successful mutations
For each bit attribute a fitness number that is proportional to the gain or loss the objective function has, compared to the best so far, if the bit is flipped
A-GEO$_1$
A-GEO$_2$
Calculate the chance of improvment CoI_i of successful flips over total flips, where success is defined by being better than the best population of all historical current populations
Calculate the chance of improvment CoI_i of successful flips over total flips, where success is defined by being better than the current population
$CoI_i = 0$
yes
Reset τ mode
no
$CoI_i \leq CoI_{i-1}$
yes
Increase τ mode
no
Make $CoI_{i-1} = CoI_i$ and Rank the bits according to their fitness numbers
Confirm the mutation of a bit of the population with probability $Pk \propto k^{-\tau}$ with $k = 1, L$
Was the stopping criteria met?
no
yes
Return the best solution found during the search

SOURCE: Made by the Author.

Improve a population on A-GEO means to find a better solution than a reference solution. To measure this improvement A-GEO adds a solution improvement measurement metric for each current generation population, called Chance-of-Improvement (CoI). CoI is calculated as

$$CoI_i = \frac{L_{\text{imp}}}{L}, \tag{3.1}$$

where L_{imp} is the number of bits in the current population that when flipped provide a better solution than a solution of reference and L is the total number of individuals (bits). If a population of ten individuals has five bits that when flipped provides better solutions than a solution of reference, the Chance of Improvement (CoI) of this population is of 0.5.

Thinking about GEO algorithm with this new variable rises a necessity to initialize its value for the first generation since, in respect to the beginning of the search, there is still no previous reference population. Thus CoI_{i-1} for $i = 0$ is given by

$$CoI_{-1} = 1/\sqrt{N}, \tag{3.2}$$

following the same rule adopted for uncorrelated mutation in evolutionary strategies (EIBEN; SMITH, 2003).

Also τ must be initialized. Any value of τ may be used to initialize the search. It may even be set randomly. Nevertheless, in the present work τ was always initiated with its value set to 0.5.

Afterwards the initialization, the CoI_i calculation occurs at each generation i, after the bit flips and before the ranking process performed by A-GEO. After it is computed A-GEO evaluates how the τ value will be changed based on three modes:

$$\text{If} \begin{cases} CoI_i = 0, \ \textbf{Re-start} \ \tau \text{ mode.} \\ CoI_i \leq CoI_{i-1}, \ \textbf{Increase} \ \tau \text{ mode.} \\ \text{otherwise, } \tau \text{ stays as it is.} \end{cases} \tag{3.3}$$

In the **Re-start** and **Increase** mode τ values change based on different rules because of different goals. The first one tries to enhance the exploration performed by the algorithm reseting the value of τ to a low value, which leads to a more stochastic behavior, facilitating jumps on the search space. To perform the mutation on τ for this mode, the following equation inspired on the rule adopted for uncorrelated mutation in evolutionary strategies (EIBEN; SMITH, 2003) was used:

$$\tau = 0.5 Lognormal(0,\, 1/\sqrt{N}) \tag{3.4}$$

However, if the **Increase** mode is applied to the algorithm, τ changes by

$$\tau = \tau + (0.5 + CoI_i)U(0,1), \tag{3.5}$$

where $U(0,1)$ is a variable with uniform distribution in the interval $[0,1]$. This uniform probability distribution is used to maintain a stochastic characteristic on the quality attribution of the τ parameter control technique while using the current population CoI (CoI_i) to insert population feedback into the mutation, as seen in some techniques reviewed by Aleti and Moser (2016).

The only difference between A-GEO$_1$ and A-GEO$_2$ is the reference population used to calculate the CoI_i. For A-GEO$_1$ the best population of all historical current populations is used. At each generation a individual is chosen to be mutated and one of the new populations generated by this process is chosen to be the next generation current population, as in GEO. It's important to notice that the reference population **is not** the best population found so far, since only one of the mutation generated populations is selected to be the next current population.

In A-GEO$_2$ case, the reference population is the generation current population, it means that, for A-GEO$_2$, the CoI of a population is the number of flips that improve the current population itself over the total number of mutations.

3.2 A-GEO parameter control mechanism classification

As pointed out in Section 1.2, Karafotias et al. (2015) discussed two classification for parameter control mechanisms. The first one was in respect to how they adapt an EA parameter (deterministic, adaptive or self-adaptive control). The second classification was in respect to how the mechanism was designed (parameter specific, ensemble or parameter independent). A-GEO, as its name suggests, uses an adaptive control technique, that is responsible to control the GEO selection parameter τ. Therefore, in respect to its design classification it was conceived as a parameter specific control mechanism, being in the first quadrant of Figure 1.3.

However, the control mechanism employed is not dependent on any behavior or peculiarity presented in A-GEO. This means it should be possible to apply it to others numeric parameters as well, needing minor adjustments.

Focusing on how control mechanisms chooses or generates parameter values during the search on EAs, Aleti and Moser (2016) established a framework for parameter control techniques, as described in Section 1.2. It consists of four non-obligatory stages: feedback collection, effect assessment, quality attribution and parameter update. Thus, a parallel was traced between the adaptive parameter control technique proposed to A-GEO and the framework developed by Aleti and Moser (2016).

To understand how the proposed technique utilizes this framework lets break it into parts. First lets just analyze how it changes the algorithm by first measuring the CoI at each generation. To compute CoI, as explained earlier, the technique needs feedback of the population, in particular the solution value. After gathering the feedback, it estimates the value of CoI, by first estimating how good a solution is relative to a reference solution, what characterizes effect assessment. Next, when it finally computes the CoI, it attributes a quality to the parameter for the next generation. For last, the parameter value is updated using a deterministic update strategy.

Now knowing how A-GEO$_1$ and A-GEO$_2$ work it is possible to say that they diverge just in one stage of the framework, the effect assessment stage. Table 3.1 shows how they can be classified in each framework stage.

Table 3.1 - A-GEO classification on Aleti and Moser (2016) parameter control framework.

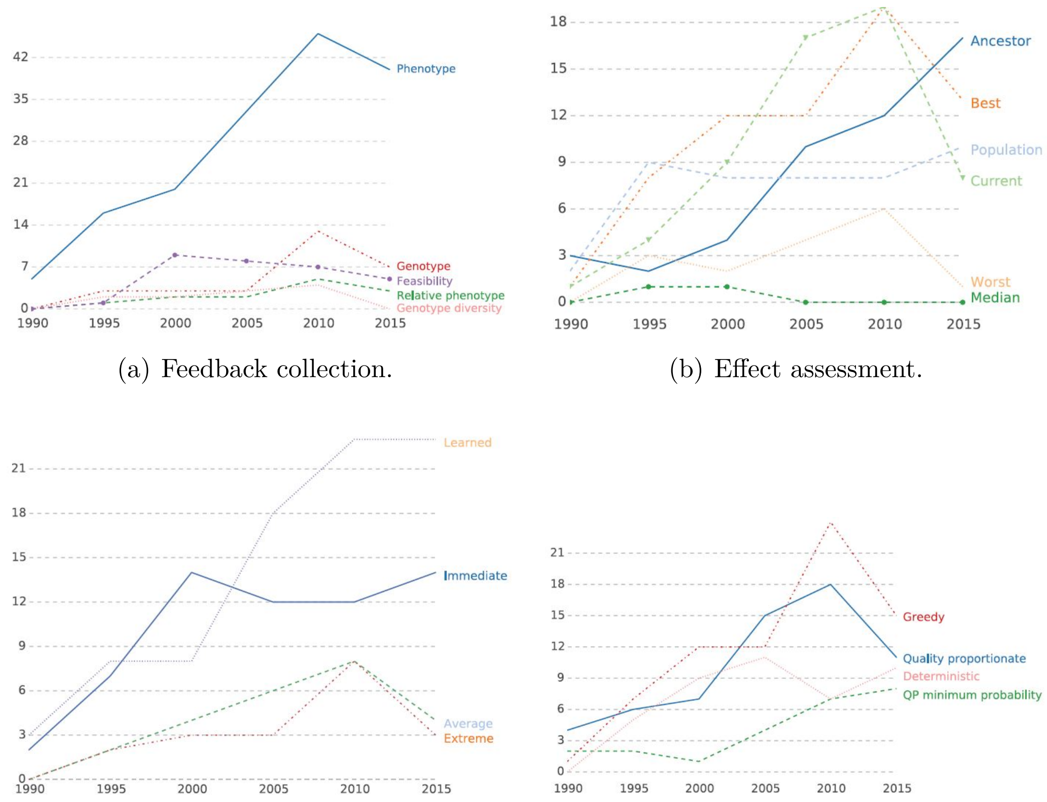
Framework stage	A-GEO$_1$	A-GEO$_2$
Feedback collection	Phenotype feedback	Phenotype feedback
Effect assessment	Best effect	Current effect
Quality attribution	Average quality	Average quality
Parameter update	Deterministic update	Deterministic update

SOURCE: Made by the author.

In respect to quality attribution and parameter update, the classification can be somewhat confusing since A-GEO uses data from the current population to attribute the parameter quality. However, remember that it uses current population data in respect to the previous one, what characterizes a subset of populations, thus an average quality attribution. In the parameter update case it is deterministic due to the parameter quality indicating exactly what will happen to the parameter value, what in A-GEO means that the quality attribution selects the mode in which the algorithm mutates τ.

Aleti and Moser (2016) also classified the papers reviewed in their research based on what features of their framework were implemented in each stage, presented in Figure 3.2. With this information its possible to position the A-GEO algorithm in this chart and verify where it stands on todays scientific community developed control mechanisms.

Figure 3.2 - Papers classified based on the adaptive parameter control mechanisms.



(a) Feedback collection.

(b) Effect assessment.

(c) Quality attribution.

(d) Parameter update.

SOURCE: Adapted by the author from Aleti and Moser (2016).

A-GEO control mechanism follows the mainstream choice of the community for feedback collection and phenotype feedback. For effect assessment A-GEO$_1$ uses the second most used choice, the best population, which until 2013 was the preferable choice, being replace by the ancestor strategy, used by A-GEO$_2$. In respect to quality attribution and parameter update A-GEO uses unpopular strategies, which is interesting since few works were conduct in the area.

This concludes the presentation of A-GEO, the motivation behind it and parallel with the bibliography reviewed. In the next chapter more insights on A-GEO are given with an analysis on how its control mechanism affects τ during the search and by performance comparisons with GEO.

4 PERFORMANCE EXPERIMENTS AND ANALYSIS

A set of test functions was chosen to evaluate the performance of A-GEO in finding their global optimum, when compared to the canonical GEO. These functions will help establish a base of comparison for controlling techniques versus tunning parameters performance, and are also used to assess the A-GEO overall performance, robustness of initialization and τ behavior.

The performance comparison for controlling versus tunning is done based on the number of function evaluations (NFE) performed given a stop criterion for all test functions except Rastrigin, which is not used for these experiments. For A-GEO overall performance and robustness of initialization experiments it is used the value of the objective function at the end of the algorithm given a large time limit. Finally τ analysis evaluates the changes in τ value during each generation of the algorithms.

Five different continuous functions, shown in Table 4.1 and Figure 4.1, were chosen. Three are separable functions (F1, F4 and F5) and two non separable (F2 and F3). F1 is DeJong #1 function, a unimodal function with three design variables. It has a global minimum at $X = \{0, 0, 0\}$ where the value of the objective function is zero.

Table 4.1 - Test functions for performance experiment.

Functions	Restrictions
F1: $f(X_n\|_{n=1,3}) = \sum_{n=1}^{N} X_n^2$	$X_n \in [-5.12; 5.12]$
F2: $f(X_n\|_{n=1,2}) = \sum_{n=1}^{N-1}[100(X_n^2 - X_{n+1})^2 + (1 - X_n)^2]$	$X_n \in [-2,048; 2,048]$
F3: $f(X_n\|_{n=1,10}) = 1 + \sum_{n=1}^{N} \frac{x_n^2}{4000} - \prod_{n=1}^{N} \cos\left(\frac{X_n}{\sqrt{n}}\right)$	$X_n \in [-600; 600]$
F4: $f(X_n\|_{n=1,20}) = 3.0N + \sum_{n=1}^{N}(X_i^2 - 3.0\cos(2\pi X_n))$	$X_n \in [-5.12, 5.12]$
F5: $f(X_n\|_{n=1,10}) = 418.9829N + \sum_{n=1}^{N}(X_n \mathrm{sin}\sqrt{\|X_n\|})$	$X_n \in [-500; 500]$

SOURCE: Made by the author.

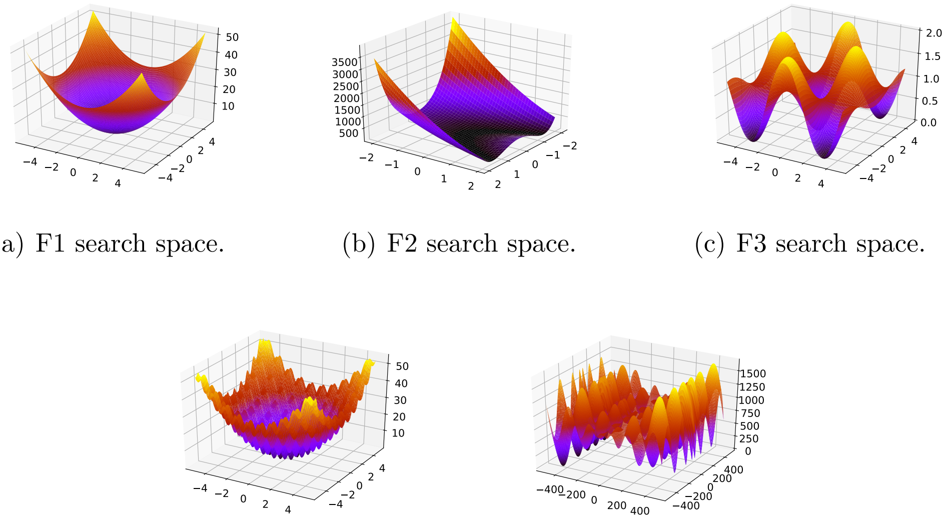
The second function, which is also a unimodal function, is known as the Rosenbrock function. It was used with two design variables and has a global minimum at $X = \{1, 1\}$ where the value of the objective function is zero.

Griewangk is the third function and has multiple local minimums, it was considered with ten design variables and has a global minimum at $X = \{0, \ldots, 0\}$ where the value of the objective function is zero.

F4 is the Rastringin function, a non-linear, separable with multiples local minimums regular distributed function, considered here with 20 design variables. It has a global minimum at $X = \{0, \ldots, 0\}$ where the value of the objective function is zero.

At last, the Schwefel function (F5) is considered with 10 design variables. It is a non-linear, separable with multiples local minimums distant from the global minimum function . It has a global minimum at $X = \{420.9687, \ldots, 420.0687\}$ where the value of the objective function is zero.

Figure 4.1 - Design spaces shown in two dimensions for all test functions.



(a) F1 search space.

(b) F2 search space.

(c) F3 search space.

(d) F4 search space.

(e) F5 search space.

SOURCE: Made by the author.

For most of the functions there were made 50 independent runs of the algorithms with a predetermined set of seeds, except for Rosenbrock, which ran 100 independent times due to its shape. Also, for all performed experiments A-GEO τ was initialized as 0.5.

4.1 Tunning versus controlling mechanism

The goal of this section is to present results and elaborate comparisons between GEO, A-GEO$_1$ and A-GEO$_2$ focused on the strategy and cost of tunning versus controlling the τ parameter. To establish a more fair base of comparison, according to Eiben et al. (1999), the GEO algorithm will be run with various τ simulating a tune of the algorithm.

The algorithms were tested with functions F1, F2, F3 and F5 defined in Table 4.1, where F4 was not used due to its similarity to F3. Their implementations details are shown in Table 4.2.

For the GEO tunning process, the τ's used for each function are presented in Table 4.3. These τ's and steps where selected through a revision of the literature which determined the min and max τ values for each function based on the best configuration value, the complexity of the function (the higher the complexity higher the step) and how long would it take to finish running all the runs with different steps.

Finally, the respective results of the experiments are in Table 4.4, Table 4.5, Table 4.6 and Table 4.7.

Table 4.2 - GEO, A-GEO$_1$ and A-GEO$_2$ tunning versus controlling configurations.

Function	Runs	Variable encoding	Stop criteria
F1	50	11 bits	$f(X) \leq 0.001$
F2	100	13 bits	$f(X) \leq 0.001$
F3	50	16 bits	$f(X) \leq 0.5$ or NFE$\geq$100,000
F5	50	16 bits	$f(X) \leq 600$ or NFE$\geq$100,000

SOURCE: Made by the author.

In Table 4.4 are shown the F1 function results using GEO and A-GEO. The second column is the mean NFE value spent to reach the stop criteria for the 50 runs of each algorithm. Note that in GEO case, two results are presented, one for the GEO tuning process, which is the mean NFE value needed to reach the stop criteria for the 50 runs of each algorithm, considering the GEO τ range and step presented for the function in Table 4.3. The other, is the mean NFE value of GEO with the best τ found in the tuning process, which for F1 is equal to 4.5.

Table 4.3 - GEO tunning τ configurations.

Function	τ boundaries (GEO)
F1	[0.5;5.0] with steps of 0.5
F2	[0.25;3.0] with steps of 0.25
F3	[0.25;4.0] with steps of 0.5
F5	[0.25;3.0] with steps of 0.5

SOURCE: Made by the author.

Table 4.4 - Tunning versus controlling F1 results for GEO and A-GEO.

Algorithm	Mean NFE	Std. Dev.
GEO tuning	3.085E+03	-
GEO (τ=4.5)	**3.500E+02**	**8.000E+01**
A-GEO$_1$	4.020E+02	9.600E+01
A-GEO$_2$	4.120E+02	8.900E+01

SOURCE: Made by the author.

As can be seen f rom the results presented in Table 4.4, if the best τ for GEO was known a priori, it would outperform A-GEO in F1. However, if a bad τ was chosen, using GEO would lead to a significantly poorer performance when compared to using A-GEO. This is highlighted when the values of the mean NFE found for A-GEO are compared with the ones found for GEO using the best τ and all τs.

In Table 4.5 are shown the F2 function results using GEO and A-GEO. For GEO, the best τ is equal to 1. For F2, the best performance of GEO is the worst of all algorithms and the overall mean performance of the tunning is 28 times worse than A-GEO$_1$ and 33 times worse than A-GEO$_2$.

Table 4.5 - Tunning versus controlling F2 results for GEO and A-GEO.

Algorithm	Mean NFE	Std.Dev.
GEO tuning	2.770E+05	-
GEO (τ=1)	1.079E+04	1.078E+04
A-GEO$_1$	9.774E+03	1.094E+04
A-GEO$_2$	**8.471E+03**	**8.648E+03**

SOURCE: Made by the author.

In Table 4.6 are shown the results for F3 using GEO and A-GEO. The third column shows the percentage of runs that met the precision criteria in 50 runs performed for each algorithm. Note that in the case of GEO, the best τ value is 1.25.

Table 4.6 - Tunning versus controlling F3 results for GEO and A-GEO.

Algorithm	Mean NFE	Std. Dev.	Met precision criteria
GEO tuning	6.106E+04	-	70%
GEO (τ=1.25)	1.370E+04	7.940E+03	**100%**
A-GEO$_1$	2.160E+04	3.458E+04	84%
A-GEO$_2$	**8.241E+03**	**4.489E+03**	**100%**

SOURCE: Made by the author.

Analyzing the runs for F3 it is clear that A-GEO outperforms the GEO tunning process, however A-GEO$_1$ presents an interesting behavior. It did not attend the precision criteria in 16% of the runs, what may be caused by a premature convergence. This happens because A-GEO$_1$ makes a lot of resets to find other local minimums, while A-GEO$_2$ can explore the search space in a more efficient approach without letting aside the exploitation. Finally A-GEO$_2$ can converge in a shorter period to the precision criteria, outperforming all the other algorithms presented.

In Table 4.7 are shown the F5 function results using GEO and A-GEO. Note that in the case of GEO the best τ is equal to 1.

Table 4.7 - Tunning versus controlling F5 results for GEO and A-GEO.

Algorithm	Mean NFE	Std. Dev.	Met precision criteria
GEO tuning	8.012E+04	-	28%
GEO (τ=1)	6.002E+04	**3.578E+04**	**70%**
A-GEO$_1$	7.216E+04	3.837E+04	42%
A-GEO$_2$	**5.918E+04**	3.898E+04	**70%**

SOURCE: Made by the author.

The results, shown in Table 4.7, use the same metrics as in Table 4.6. The behavior of the algorithms on this function is similar to the behavior on Griewangk since both have multiple local minimums. A-GEO outperforms GEO relatively to the tunning necessity. Also, A-GEO$_2$ outperforms all the other versions. However it does not deliver great improvements in respect to the best GEO for this deceptive function.

All these experiments suggest that A-GEO outperforms GEO as the complexity of the functions rises. This is a promising result, since high complexity functions are common in real world scenarios. Considering the necessity to have the user tunning the algorithm to retrieve a great performance from GEO requires a know how about the algorithm from the user and imposes a considerable performance deficit. Also, A-GEO$_2$ outperforms the best static configuration of GEO in almost all functions.

4.2 A-GEO performance experiments and τ analysis

GEO, A-GEO$_1$ and A-GEO$_2$ overall performance was analyzed over long runs where the algorithms searched for the best objective function value they can reach in a time limit. Meanwhile each independent search progress (best solution found so far) was stored to evaluate how different independent runs of the algorithms behave. For this porpoise the samples objective function standard deviation σ, and coefficient of variation c_{v} (LOVIE, 2005) were computed by:

$$c_{\mathrm{v}} = \frac{\sigma}{\mu}, \tag{4.1}$$

where μ is the mean value of the objective function samples. Both of them are used to measure the variability of the best solutions found through different independent runs and help develop some insights about the balance between exploration and exploration.

Also, for these runs, it was analyzed the behavior of τ along the search. The τ analysis uses as a time axis the number of generations, i, of GEO or its variants. This metric was chosen because τ mutates at each generation on A-GEO and each generation is related to the NFE, the number of variables N, and the number of bits that the design variables were encoded m by

$$\mathrm{NFE}_i = N \times m. \tag{4.2}$$

Where NFE_i is the number of function evaluations that occur in generation i. To illustrate how it works, based on Equation (4.2), for F1, that has a stop criteria of $i \geq 37$, in terms of NFE it would be greater than 1,200.

Both experiments are executed for each test function presented in Table 4.1, and their configurations are presented in Table 4.8. The GEO algorithm used for comparison is the one with best τ statical configuration, presented in Section 4.1.

Table 4.8 - GEO, A-GEO$_1$ and A-GEO$_2$ overall performance general configurations.

Function	Runs	Variable encoding	Stop criteria
F1	50	11 bits	NFE$\geq$1.200E+03 (i $\geq$3.700E+01)
F2	100	13 bits	NFE$\geq$1.000E+06 (i $\geq$3.846E+04)
F3	50	16 bits	NFE$\geq$1.000E+06 (i $\geq$6.250E+03)
F4	50	16 bits	NFE$\geq$1.000E+06 (i $\geq$3.125E+03)
F5	50	16 bits	NFE$\geq$1.000E+06 (i $\geq$6.250E+03)

SOURCE: Made by the author.

4.2.1 Test Function 1 (DeJong #1)

The performance results for this function are shown in Figure 4.2. Analyzing first the graphs presented for performance, it can be stated that GEO has the best overall performance. Also, GEO demonstrated to present almost the same disparity between its best solution values found through the search as the other algorithms, as shown by the coefficient of variation and standard deviation.

The different solutions presented by each independent run of A-GEO$_2$ start to strongly diverge in value between 400 and 600 NFE, which can be caused by reset mutations on the values of τ that leads the algorithm to other search spaces. After some more evaluations it normalizes its c_v and converge with the other algorithms.

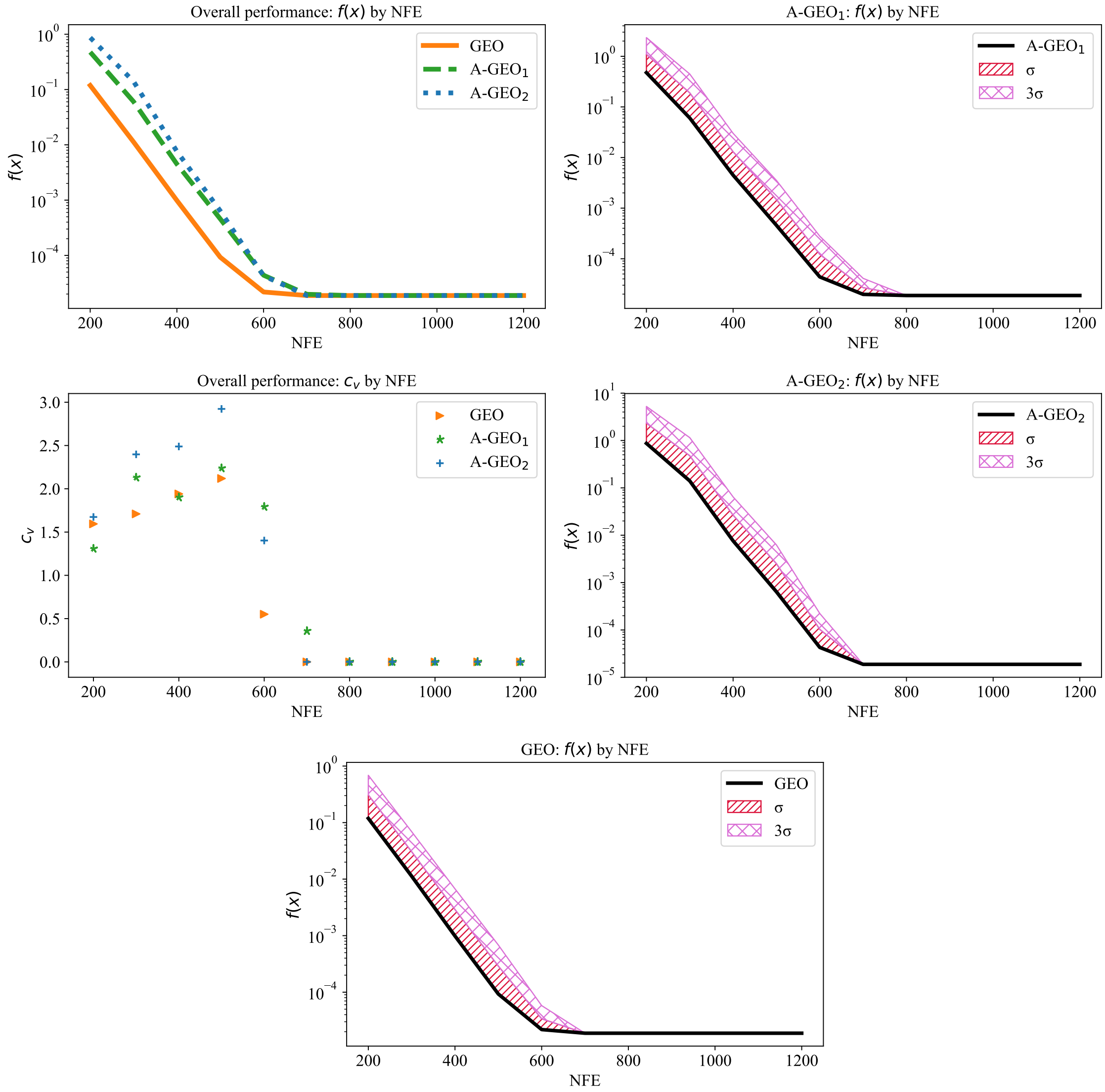
The evolution of τ, shown in Figure 4.3 for all algorithms, demonstrate a strong τ value increase in the first twelve generations for A-GEO$_.$ It can be noticed that around generation thirteen, which is close to NFE 400, the τ value starts to stagnate and then decrease. This can cause a variation on the c_v of the objective function, since some runs that start to decrease the parameter value in earlier generations are performing resets while others can be maintaining their τ value through some more iterations.

Further in the search, the first approach stabilizes τ between 0.4 and 0.8 while the second one sees an increase on it again. This stabilization means that the τ

parameter suffers fewer increases and more resets along the next generations or it is maintaining its τ value. To analyze this last statement an average of the data collected for the behavior of τ is presented in Table 4.9.

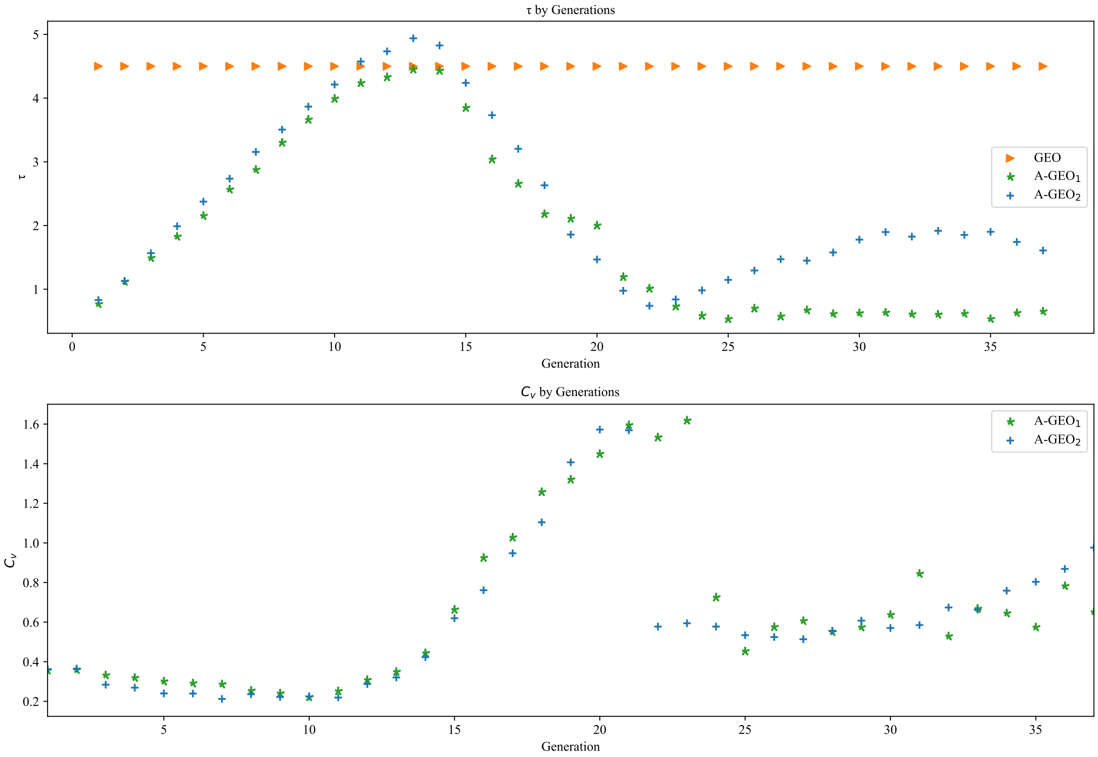
The columns on Table 4.9, starting from the second one are: the average of resets, the average of increases, the average of no changes and the average value of τ performed on the 50 independent runs of A-GEO. Analyzing the data A-GEO$_2$ has a higher τ average, almost no resets, more increases and maintained τ's more times than A-GEO$_1$, what may imply that it balances better its exploitation and exploration.

Figure 4.2 - Overall performance F1 results for GEO and A-GEO.



SOURCE: Made by the author.

Figure 4.3 - F1 function evolution of τ parameter through generations.



SOURCE: Made by the author.

Table 4.9 - F1 τ average results.

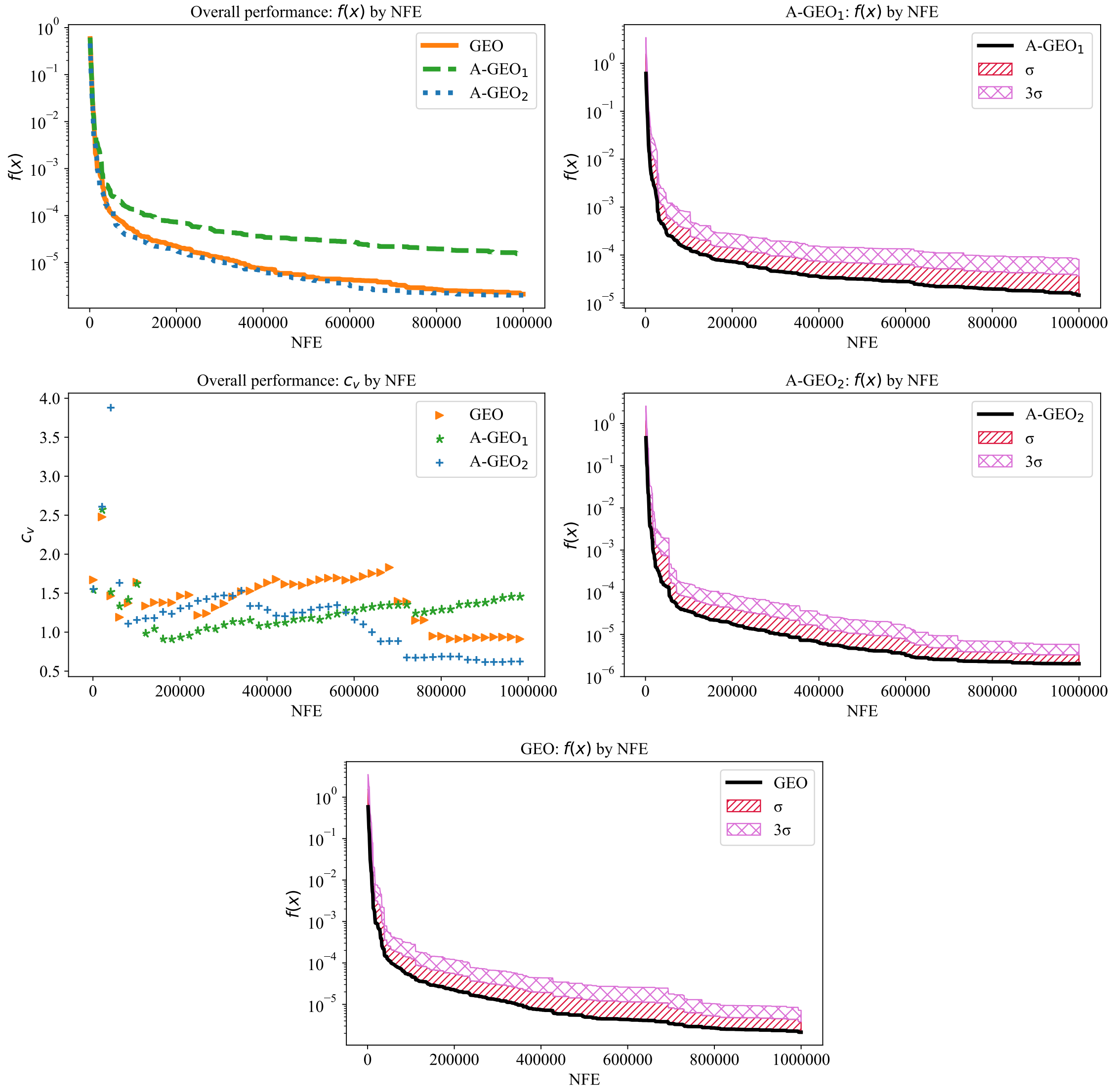
Algorithm	τ Reseted	τ Increased	τ Maintained	$\overline{\tau}$
A-GEO$_1$	2.100E+01	1.500E+01	1.000E+00	1.850E+00
A-GEO$_2$	2.000E+00	2.400E+01	1.200E+01	2.340E+00

SOURCE: Made by the author.

4.2.2 Test Function 2 (Rosenbrock Function)

The performance results for this function are shown in Figure 4.4. Looking at the graphs its clearly that A-GEO and GEO have similar performance, with A-GEO$_2$ being the best among the three. When it comes to their disparity between solution values through different independent runs, A-GEO variations have the highest through almost all the search.

Figure 4.4 - Overall performance F2 results for GEO and A-GEO.

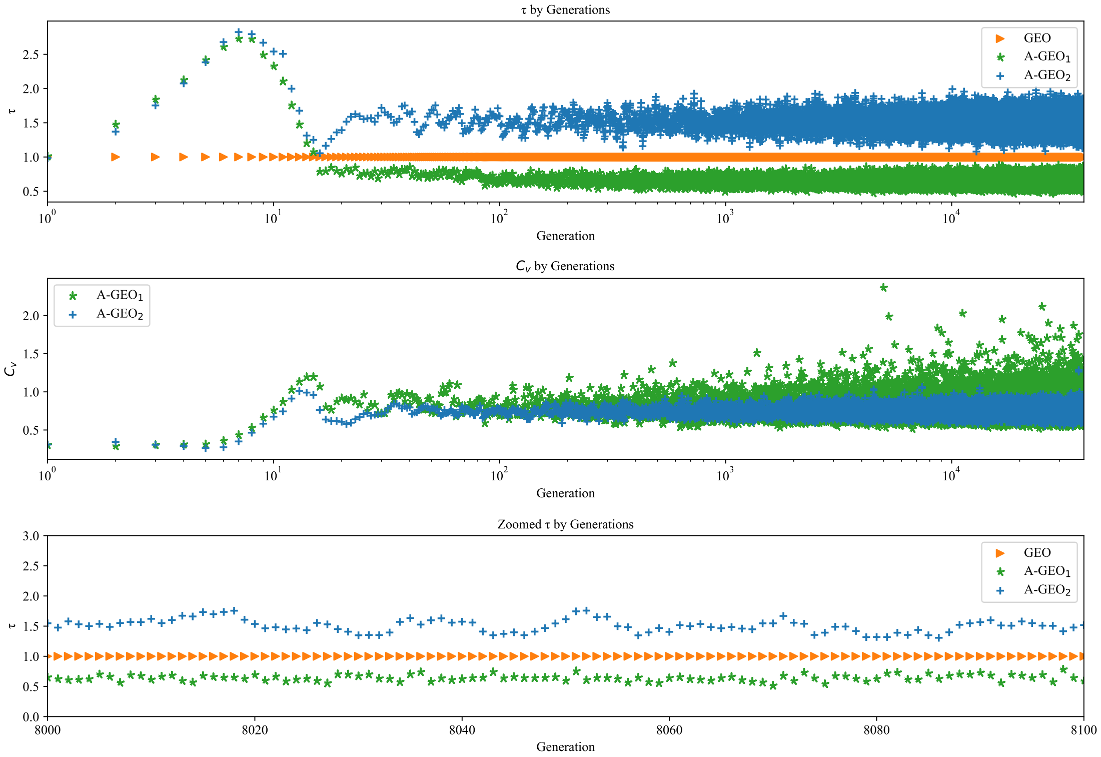


SOURCE: Made by the author.

The evolution of τ through the search in A-GEO, shown in Figure 4.5, first increases its value in the first ten generations. Rapidly after this, A-GEO$_1$ drops its τ value to less than the best τ of GEO due to a lot of resets taken. It is important to notice that A-GEO$_2$ maintain its τ values always higher than the values presented by GEO, while A-GEO$_1$ after the first increase and decrease cycle does not reach GEO best τ. Analyzing those values as the search pass through the generations it is observed that the τ values amplitude increases as the search advances. This can be caused by the difficult in improving a solution as it approaches the global optimum value forcing resets on the algorithm. Also, the c_v by Generations plot demonstrates that

A-GEO$_2$ τ mutations are more consistent through different runs.

Figure 4.5 - Rosenbrock function evolution of τ for A-GEO$_1$ and A-GEO$_2$.



SOURCE: Made by the author.

After the abruptly increase and decrease of τ value, A-GEO$_2$ starts to better balance the search in terms of exploitation and exploration making it find better solutions than the other approaches. The overall behavior of τ can be summarized by Table 4.10, showing that A-GEO$_2$ may be a more balanced approach than A-GEO$_1$ since it presents an equilibrium between the number of times that τ values increases and are maintained. Also, other factor that contributes to the low disparity between solution values on the different runs, and also to A-GEO$_2$ being a better balanced algorithm is that this function has a single valley, which as can be seem was better exploited by A-GEO$_2$.

Table 4.10 - Rosenbrock τ average results.

Algorithm	τ Resets	τ Increases	τ Maintained	$\overline{\tau}$
A-GEO$_1$	3.842E+04	2.200E+01	2.400E+01	6.400E-01
A-GEO$_2$	3.681E+03	1.942E+04	1.536E+04	1.510E+00

SOURCE: Made by the author.

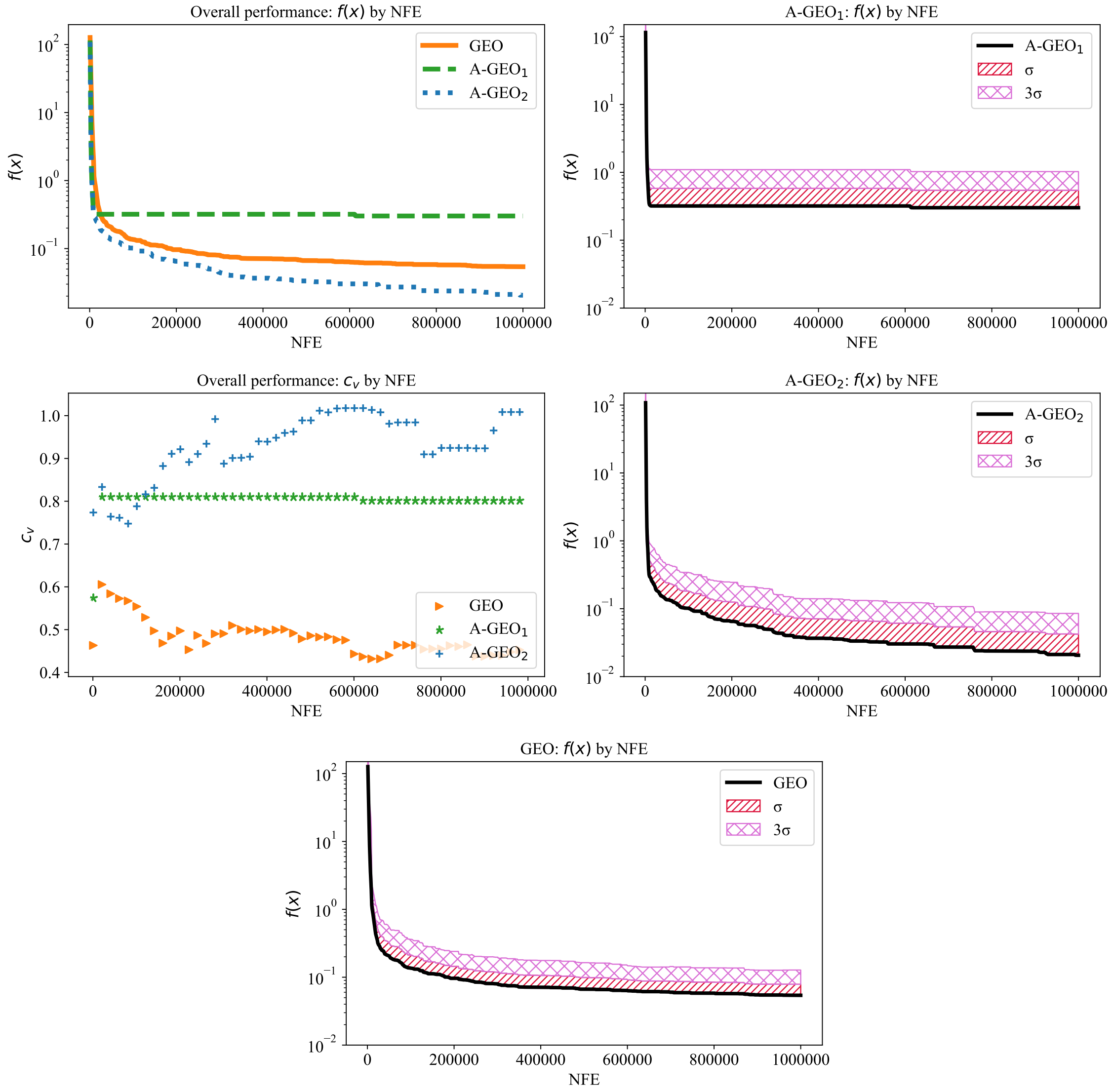
4.2.3 Test Function 3 (Griewangk Function)

The performance results for this function are shown in Figure 4.6. In this case A-GEO$_1$ is beaten by all the algorithms in performance, while A-GEO$_2$ clearly suppress GEO. However, analyzing the oscillations of the solution values, found by the independent runs, GEO demonstrates to find extremely likely solution values on its independent runs during all the search while A-GEO$_2$ increases its disparity as the search advances.

This behavior of A-GEO$_2$ can be justified by the adaption of its free parameter during the search, that generates better results, but sacrifices solution convergence in doing so. This behavior is indication of a trade off between exploration and exploitation. A possible explanation is that since τ can be reseted making the algorithm explore new search spaces when it stagnates, each independent run has bigger chances to be on different search spaces at each generation. After a new promising location is found, the search starts an exploitation cycle that starts to make the algorithm found more likely solution values, thus decreasing its c_v. However, experiments that analyzes the position that the algorithm population finds itself on the search space, at each generation, have to be made to prove this possibility.

The adaptive parameter evolution results are shown in Figure 4.7. A-GEO$_1$ and A-GEO$_2$ increase the average τ value in the first forty generations. However, after this A-GEO$_1$ starts to reset it constantly, due to the lack of capacity to improve the population value in respect to the best population found so far. This does not mean that A-GEO$_2$ does not perform those resets too, however they are less frequent as can be seen in Table 4.11.

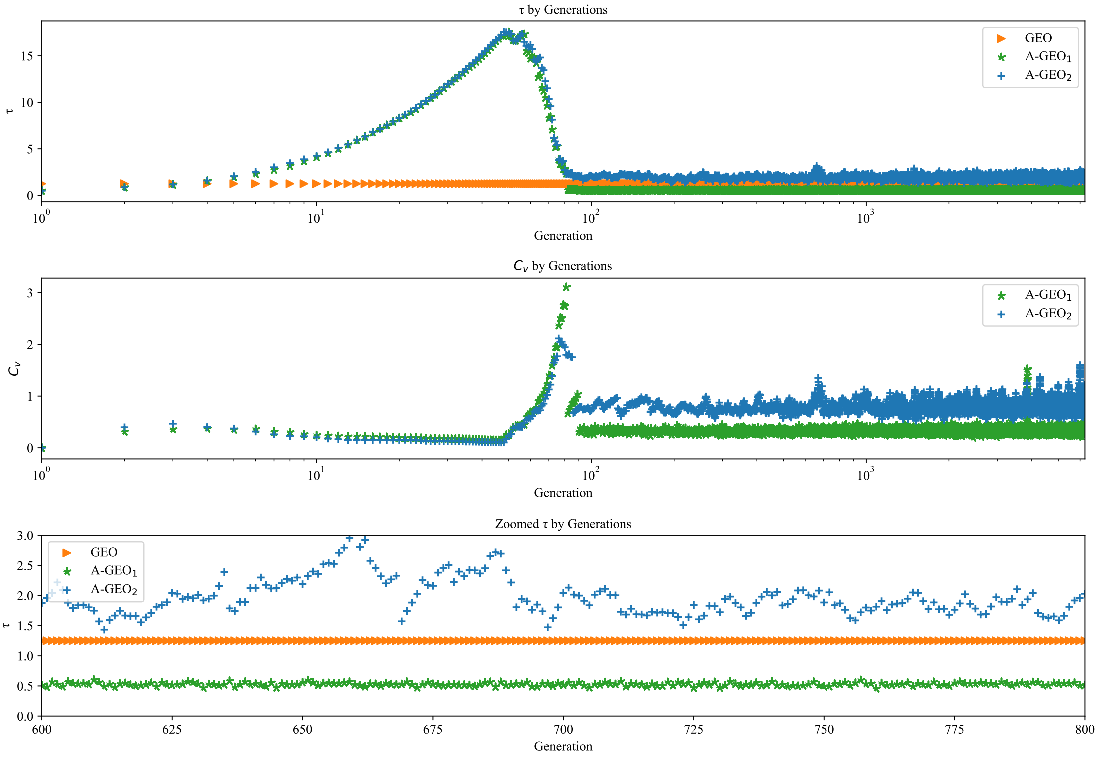
Figure 4.6 - Overall performance F3 results for GEO and A-GEO.



SOURCE: Made by the author.

After getting to the balance point, close to 0.65 for A-GEO$_1$ and two for A-GEO$_2$, shown in Table 4.11, A-GEO$_2$ increase and decrease cycles of τ have very different sizes through the evolution of the search. This can mean that the independent runs are finding different solutions because they have different values of τ, which leads to different points on the search space, or vice-versa. This possibility is reinforced by the c_v presented in Figure 4.6 for the objective function value during the search and by the c_v of τ values in Figure 4.7. Thus, this can result in different regions on the search space since greater c_v implies a bigger distance between values.

Figure 4.7 - Griewangk function evolution of τ for A-GEO$_1$ and A-GEO$_2$.



SOURCE: Made by the author.

Table 4.11 - Griewangk τ average results.

Algorithm	τ Reseted	τ Increased	τ Maintained	$\overline{\tau}$
A-GEO$_1$	6.182E+03	6.000E+01	8.000E+00	6.500E-01
A-GEO$_2$	3.530E+02	3.671E+03	2.226E+03	2.060E+00

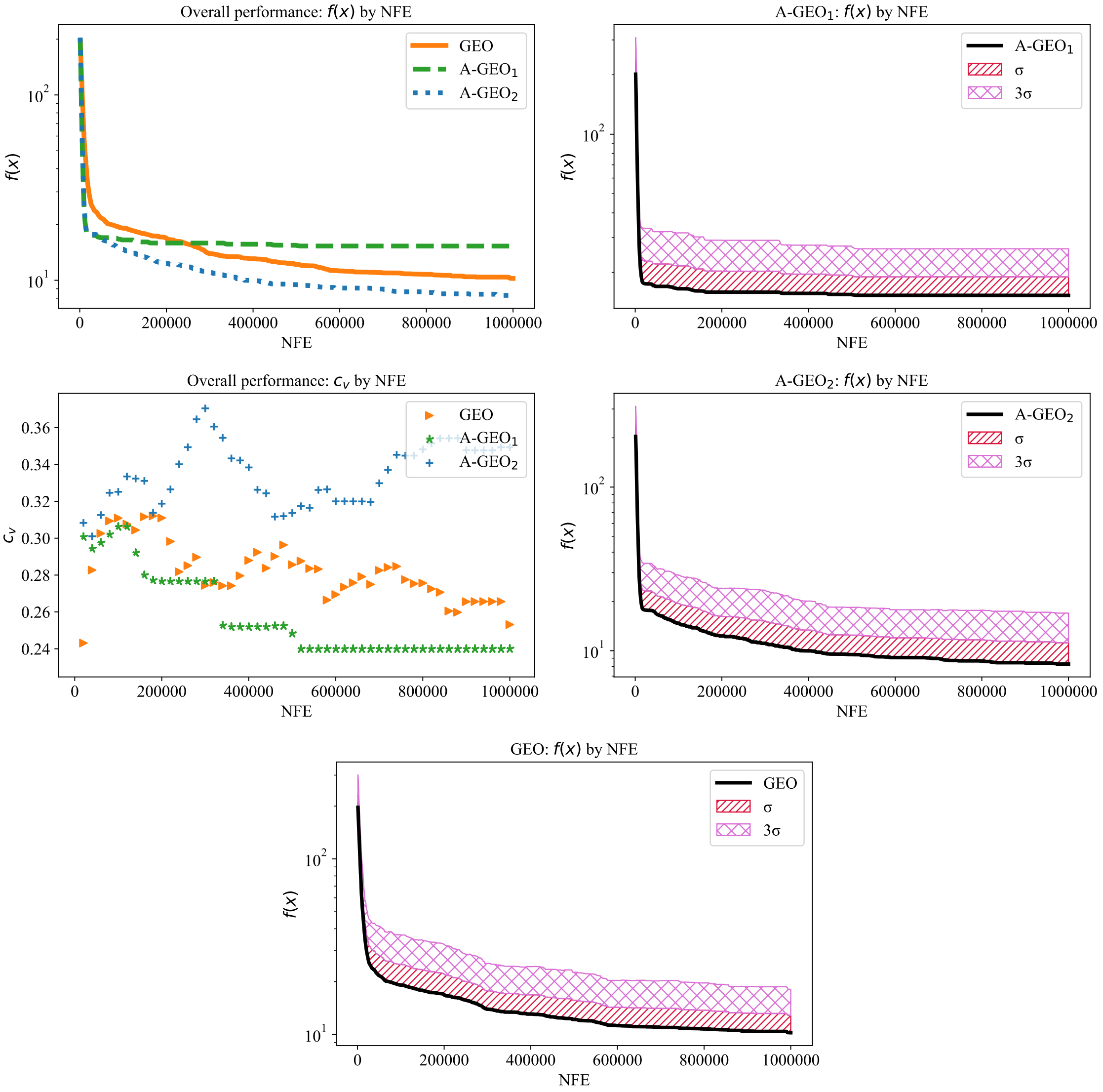
SOURCE: Made by the author.

Considering that the second version of the algorithm performs the adaption of τ based on the current and possible future populations, the chances to make τ more deterministic or maintain it in a short period increases, as shown by the zoomed graph in Figure 4.7, where the increasing parts of the graph for the second algorithm persist more time than the ones of the first one.

4.2.4 Test Function 4 (Rastringin Function)

Similar to the Griewangk results A-GEO$_2$ outperforms all approaches, as shown in Figure 4.8. However, it does not present a relatively high c_v compared to GEO as in Function F3. In A-GEO$_1$ case a decreasing c_v is also observed, but it is probably due to premature convergence to a local optimum, from which it can not scape. This can be reinforced by Figure 4.9, which shows the adaptive parameter evolution results.

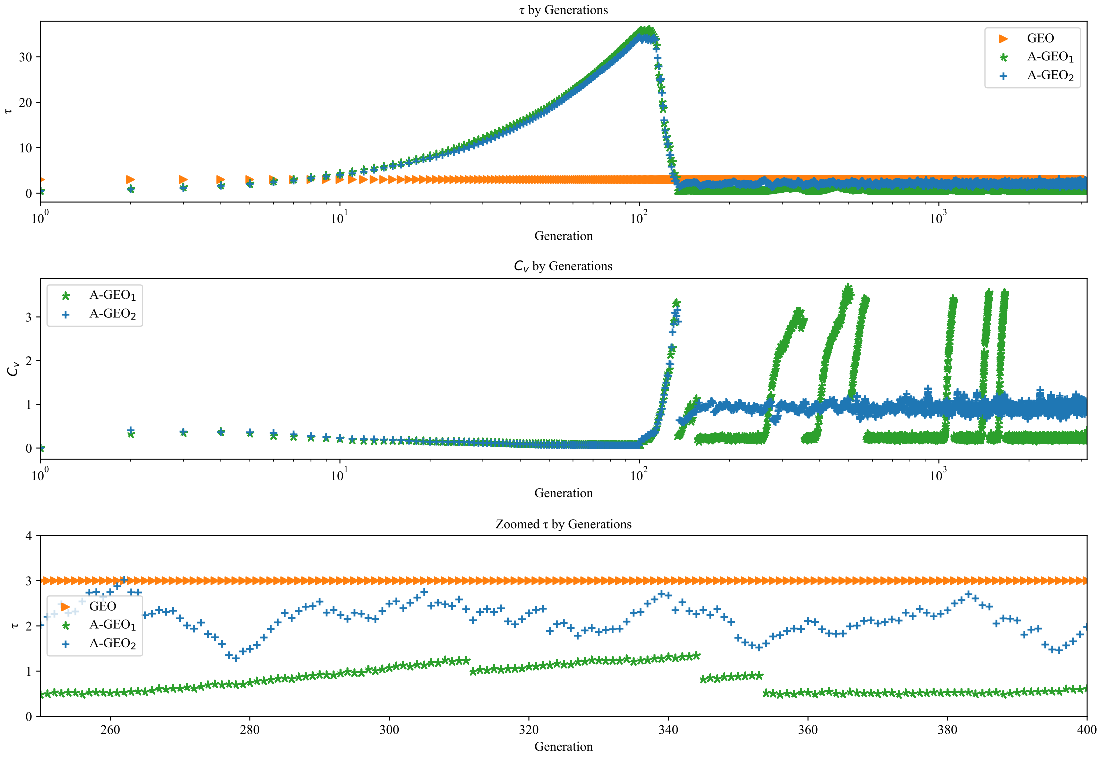
Figure 4.8 - Overall performance F4 results for GEO and A-GEO.



SOURCE: Made by the author.

The evolution of τ for both A-GEO$_1$ and A-GEO$_2$, increases the parameter value in the first one hundred generations. However, after this A-GEO$_1$ start to reset it constantly, due to the lack of capacity to improve the population value in respect to the best population found so far, as can be seen in Table 4.12.

Figure 4.9 - Rastringin function evolution of τ for A-GEO$_1$ and A-GEO$_2$.



SOURCE: Made by the author.

After getting to the balance point, close to 0.5 for A-GEO$_1$ and two for A-GEO$_2$, the algorithms start to diverge in behavior. A-GEO$_1$ performs cycles of slow increases to τ through some generations and then restarts it close to the balance point. A-GEO$_2$ has a more random behavior, after it reaches for the first time the balance point in respect to the cycle of τ, it increase and decrease cycles have very different sizes through the evolution of the search and the values of τ are always greater than the values of A-GEO$_1$, as it performs less resets.

Finally, Table 4.12 shows that A-GEO$_2$ has a higher average, fewer resets, more increases and maintained τ for more times than A-GEO$_1$, what may imply that it is more balanced than A-GEO$_1$. Also, the c_v of GEO can indicate that the algorithm is converging to a solution value that it can not improve while A-GEO$_2$ through its different runs is still exploring new solutions, favoring its exploration without letting the exploitation aside.

Table 4.12 - Rastringin τ average results.

Algorithm	τ Reseted	τ Increased	τ Maintained	$\overline{\tau}$
A-GEO$_1$	2.987E+03	1.300E+02	8.000E+00	1.400E+00
A-GEO$_2$	1.110E+02	1.782E+03	1.282E+03	2.830E+00

SOURCE: Made by the author.

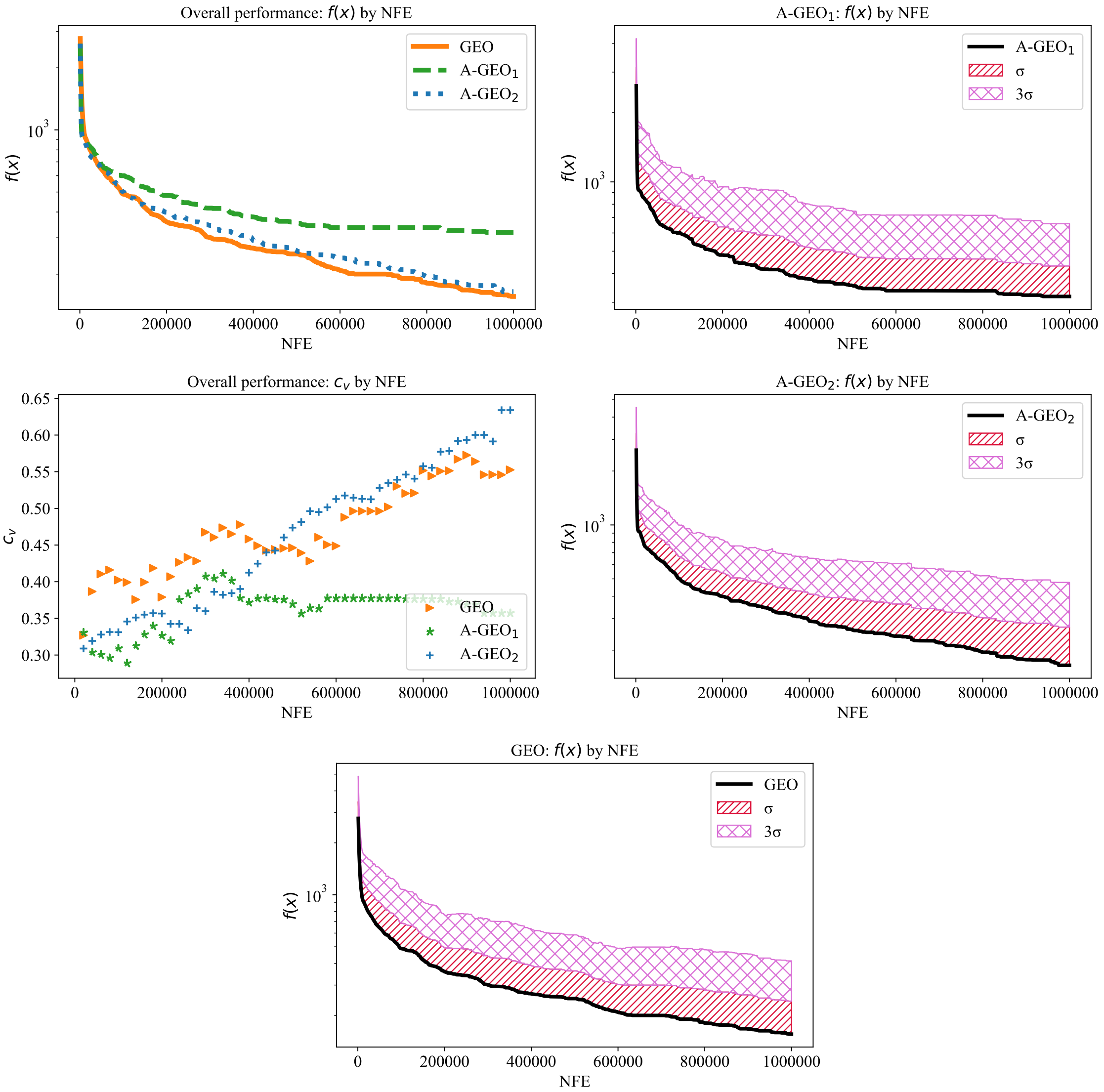
4.2.5 Test Function 5 (Schwefel Function)

The performance results for this function are shown in Figure 4.10. A-GEO$_1$ presents the lowest disparity between its runs of the three algorithms, however it seems to premature converge to a local optimum as in the Rastringin function. A-GEO$_2$ is similar to GEO in performance and solution values disparity in this function. Both present difficult to deal with a deceptive function, which caused the c_v of the algorithms to increase during the search as they look for more promising search space.

The adaptive parameter evolution results are shown in Figure 4.11. The evolution of τ for both A-GEO$_1$ and A-GEO$_2$, increases its value in the first fifth generations. Both A-GEO algorithms exhibit the same behavior for τ as in Griewangk or Rastringin, with A-GEO$_1$ performing a extremely high number of resets, while A-GEO$_2$ finds a balance between increasing and maintaining τ values.

Table 4.13 shows that A-GEO$_2$ has a higher average, less resets, more increases and maintained τ for more times than A-GEO$_1$, what may imply that it is more balanced than A-GEO$_1$.

Figure 4.10 - Overall performance F5 results for GEO and A-GEO.



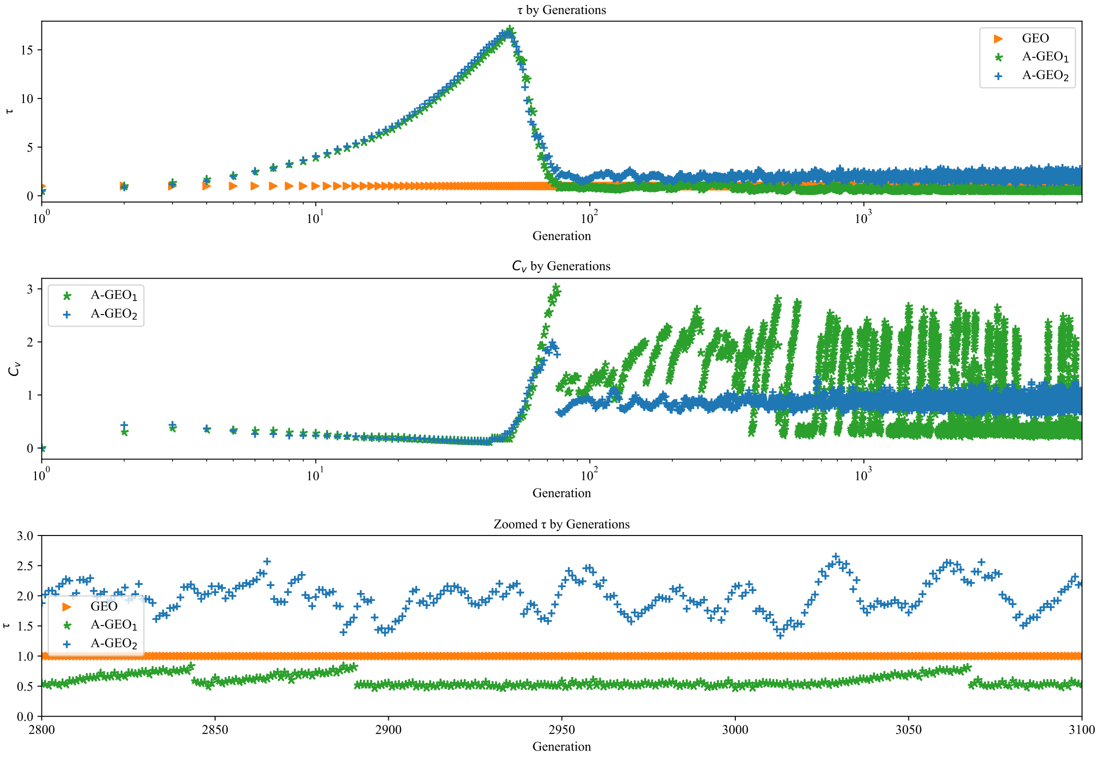
SOURCE: Made by the author.

Table 4.13 - Schwefel τ average results.

Algorithm	τ Reseted	τ Increased	τ Maintained	$\overline{\tau}$
A-GEO$_1$	6.086E+03	1.400E+02	2.400E+01	7.100E-01
A-GEO$_2$	2.850E+02	3.506E+03	2.459E+03	2.070E+00

SOURCE: Made by the author.

Figure 4.11 - Schwefel function evolution of τ for A-GEO$_1$ and A-GEO$_2$.



SOURCE: Made by the author.

From the results presented for performance on this section it can be seen that A-GEO$_1$ is no match for GEO in complex functions. However it has no free parameter to be tunned, what is an enormous advantage. In respect to A-GEO$_2$, it outperforms GEO in almost every function, possibly offering a better trade off between its exploitation and exploration than GEO.

Analyzing the behavior of τ, its clear that the initial high climbing in value lasts more as the complexity of the function increases. After τ values decrease to a balance point the A-GEO variations differ in behavior. A-GEO$_1$ present lower τ values than GEO at all functions except for F5 in some points, what can be interpreted as if GEO best τs are too deterministic at each stage of the search for A-GEO$_1$, thus τ is forced to be reseted before it reaches those values. These resets try to force the algorithm to explore other search spaces along the search to obtain better solutions than the best ones found so far.

Looking to the values of τ for A-GEO$_2$ they are always higher than the ones of A-GEO$_1$ and for some functions even higher than the best value for GEO. This is a consequence of performing less resets and maintaining the value of the free parameter through more generations when it still offers diverse better solutions to be explored. Then, when the number of better solutions gets narrowed the algorithm forces exploitation by increasing τ value.

As pointed out before, it can be seen that the resets and no changes in τ lower its average value in some stages of the algorithm. This can occur when some runs are searching for new exploration spaces while others are still exploiting their current regions. Thus, when the τ in different independent runs differ on behavior, the c_v increases.

In summary, the results show that making GEO adaptive can not only free its user of the burden of its free parameter τ, but also provide better results, even if the "best" τ for GEO is know a priori. In the next Chapter A-GEO continues to be analyzed an extremely more complex test suite.

4.3 A-GEO: A Glance Into the Future

A-GEO is a new version of GEO, an almost 20 years old evolutionary algorithm. Since then a lot has changed in the field of evolutionary computation. As mentioned through this work, there are a lot of new evolutionary algorithms with adaptive and self-adaptive versions, and some of them are even using artificial intelligence and machine learning techniques to improve their performance. All those algorithms performed a long path being evolved till their current version to be considered state of the art evolutionary algorithms.

It is not expected of A-GEO to be fully competitive with these algorithms, however comparing it with them can give useful insights on how close A-GEO brings GEO to them and where are the opportunities for improvement.

Thus, in this section A-GEO$_2$ is compared with a recent evolutionary algorithm named Teaching Learning Based Optimization with Focused Learning (TLBO-FL) over 30 benchmark functions from the 2017 IEEE Congress of Evolutionary Computation (CEC2017).

This algorithm was chosen due to being the worst ranked algorithm presented in the CEC2017 competition, and it serves as a kind of metric for the performance of A-GEO$_2$ when compared to todays evolutionary algorithms.

First the benchmarks of CEC2017 are presented, followed by a brief description of the TLBO-FL algorithm, and finally the results and comparisons for both algorithms are shown.

4.3.1 2017 IEEE Congress of Evolutionary Computation Benchmark

Every year, since 2005, IEEE Congress of Evolutionary Computation makes a new competition on real parameter single objective optimization (AWAD et al., 2017). All the selected algorithms are tested in 30 single objective benchmark functions that can be transformed into dynamic, niching composition, computationally expensive and many other classes of problems (AWAD et al., 2017).

The CEC2017 benchmark is composed of 30 bound constrained real parameter single objective problems, shown in Table 4.14. Problems B1 to B3 are unimodal, B4 to B10 are multimodal, B11 to B20 are hybrid and B21 to B30 are composition functions. Each function is shifted and rotated by a different matrix, o and M respectively (AWAD et al., 2017). Also, they have D dimensions with search range $[-100, 100]$ (AWAD et al., 2017).

For this benchmark the hybrid functions are defined by Awad et al. (2017, p. 16) as:

$$\begin{aligned}
F(x) &= g_1(M_1 z_1) + g_2(M_2 z_2) + \ldots + g_N(M_N z_N) + F^*(x) \\
F(x) &: \text{ hybrid function} \\
g_i(x) &: \ i^{th} \text{ basic function used to construct the hybrid function} \\
N &: \text{ number of basic functions} \\
z &= [z_1, z_2, \ldots, z_N] \\
z_1 &= [y_{S_1}, y_{S_2}, \ldots, y_{S_{n_1}}], z_2 = [y_{S_{n_1+1}}, y_{S_{n_1+2}}, \ldots, y_{S_{n_1+n_2}}], \\
z_N &= [y_{S_{\sum_{i=1}^{N=1} n_{i+1}}}, y_{S_{\sum_{i=1}^{N=1} n_{i+2}}}, \ldots, y_{S_D}] \\
y &= x - o_i, S = \text{randperm}(1:D) \\
p_i &: \text{ used to control the percentage of } g_i(x) \\
n_i &: \text{ dimension for each basic function } \sum_{i=1}^{N} n_i = D \\
n_1 &= [p_1 D], n_2 = [p_2 D], \ldots, n_{N-1} = [p_{N-1} D], n_N = D - \sum_{i=1}^{N-1} n_i
\end{aligned}$$

Next, the composition functions are the combinations of basic functions that merge the properties of the sub-functions and maintains continuity around the global optima (AWAD et al., 2017). They were defined by Awad et al. (2017, p. 19) as:

$$F(x) = \sum_{i=1}^{N} \{\omega_i * [\lambda_i g_i(x) + bias_i]\} + F*$$

$F(x)$: composition function

$g_i(x)$: i^{th} basic function used to construct the hybrid function

N : number of basic functions

o_i : new shifted optimum position for each $g_i(x)$, define the global and local optima's position

$bias_i$: defines which optimum is global optimum

σ_i : used to control each $g_i(x)$'s coverage range, a small σ_i gives a narrow range for that $g_i(x)$

λ_i : used to control each $g_i(x)$'s height

w_i : weight value for each gi(x), calculated as below:

$$w_i = \frac{1}{\sqrt{\sum_{j=1}^{D}(x_j - o_{ij})^2}} \exp - \frac{\sum_{j=1}^{D}(x_j - o_{ij})^2}{2D\sigma_i^2}$$

Then normalize the weight $w_i = w_i / \sum_{i=1}^{n} w_i$

So when $x = o_i, w_j = \left\{ \begin{array}{cc} 1 & j = i \\ 0 & j \neq i \end{array} \right\}$ for $j = 1, 2, \ldots, N, f(x) = bias_i + f^*$

The local optimum which has the smallest bias value is the global optimum. The composition function merges the properties of the sub-functions better and maintains continuity around the global/local optima. Functions $F_i' = F_i - F_i^*$ are used as g_i. In this way, the function values of global optima of g_i are equal to 0 for all composition functions in this report. In CEC'14, the hybrid functions are also used as the basic functions for composition functions (Composition Function 7 and Composition Function 8). With hybrid functions as the basic functions, the composition function can have different properties for different variables subcomponents.

4.3.2 Teaching Learning Based optimization with Focused Learning (TLBO-FL)

TLBO-FL is a variant of TLBO (RAO et al., 2011), an evolutionary algorithm that implements a population based meta heuristic technique to mimic the knowledge transfer in a classroom through teaching and learning process (Kommadath; Kotecha, 2017; RAO et al., 2011).

Each design variable X_n is considered a subject for each student X, represented as a solution in the population (Kommadath; Kotecha, 2017; RAO et al., 2011). There are two phases in the algorithm: teaching phase and learning phase. Every student undergoes both phases, and the teacher X^{best} is the solution with the best fitness in the population (Kommadath; Kotecha, 2017; RAO et al., 2011). During the teaching

Table 4.14 - Summary of the CEC2017 Test Functions.

No.	Functions	$F_i^* = F_i(x^*)$
B1	Shifted and Rotated Bent Cigar Function	100
B2*	Shifted and Rotated Sum of Different Power Function	200
B3	Shifted and Rotated Zakharov Function	300
B4	Shifted and Rotated Rosenbrock's Function	400
B5	Shifted and Rotated Rastrigin's Function	500
B6	Shifted and Rotated Expanded Scaffer's F6 Function	600
B7	Shifted and Rotated Lunacek Bi Rastrigin Function	700
B8	Shifted and Rotated Non-Continuous Rastrigin's Function	800
B9	Shifted and Rotated Levy Function	900
B10	Shifted and Rotated Schwefel's Function	1000
B11	Hybrid Function 1 (N=3)	1100
B12	Hybrid Function 2 (N=3)	1200
B13	Hybrid Function 3 (N=3)	1300
B14	Hybrid Function 4 (N=4)	1400
B15	Hybrid Function 5 (N=4)	1500
B16	Hybrid Function 6 (N=4)	1600
B17	Hybrid Function 6 (N=5)	1700
B18	Hybrid Function 6 (N=5)	1800
B19	Hybrid Function 6 (N=5)	1900
B20	Hybrid Function 6 (N=6)	2000
B21	Composition Function 1 (N=3)	2100
B22	Composition Function 2 (N=3)	2200
B23	Composition Function 3 (N=4)	2300
B24	Composition Function 4 (N=4)	2400
B25	Composition Function 5 (N=5)	2500
B26	Composition Function 6 (N=5)	2600
B27	Composition Function 7 (N=6)	2700
B28	Composition Function 8 (N=6)	2800
B29	Composition Function 9 (N=3)	2900
B30	Composition Function 10 (N=3)	3000

*Notice that exceptionally in CEC2017 the problem B2 has been excluded due to unstable behavior and performance variations for the same algorithm (AWAD et al., 2017).

SOURCE: Adapted by the author from Awad et al. (2017).

phase the professor tries to improve the class mean fitness while in the learning phase the student try to improve itself by pairing with another random student $X^{partner}$ (Kommadath; Kotecha, 2017; RAO et al., 2011). In both phases a new solution is accepted only if it is better than the previous one (Kommadath; Kotecha, 2017; RAO et al., 2011). Finally, each solution pass through the teaching and learning phase before another repeats the process.

In TLBO-FL all students try to improve their knowledge through the teaching phase, however only the solutions that could not be improved in this phase go to the learning phase (Kommadath; Kotecha, 2017). In TLBO-FL learning phase, a student pairs itself with two other students, one having a better fitness value and the other a worse fitness (Kommadath; Kotecha, 2017).

The algorithm steps defined by Kommadath and Kotecha (2017) are to first generate a random uniform population within the bounds of the design variables and evaluate their fitness. Then a teacher is selected (X^{best}) and modified by the following equation:

$$\left.\begin{array}{ll} X_n^{newbest} = X_n^{best} + r_n X_n^{best} & if\ n \neq k \\ X_n^{newbest} = X_n^{best} & otherwise \end{array}\right\} \forall n = 1, 2, \ldots, N, \tag{4.3}$$

where X_n^{best} is the value of the n^{th} decision variable of the best solution, k is a random number between 1 and N, and r_n is a random number in the range of $[0, 1]$ generated independently for each solution (Kommadath; Kotecha, 2017). Next the solution is corrected in case of boundary violation by

$$\begin{array}{ll} X_n^{newbest} = \min(X_n^{newbest}, X_n^u) & \forall n = 1, 2, \ldots, N \\ X_n^{newbest} = \max(X_n^{newbest}, X_n^l) & \forall n = 1, 2, \ldots, N \end{array}. \tag{4.4}$$

If $X^{newbest}$ is worse than X^{best} a new potential solution is generated at the learning phase by

$$X_n^{newbest} = X_n^{best} + r_n(X_n^{best} - X_n^{partner}), \tag{4.5}$$

where $X_n^{partner}$ is the n^{th} design variable from any other solution of the solution pool. Then the rest of the students are submit to the teaching phase (Kommadath; Kotecha, 2017), by the following equation:

$$X_n^{new} = X_n + r_n(X_n^{best} - T_F X_n^{mean}), \tag{4.6}$$

where X_n^{mean} is the mean of the current class in the n^{th} design variable and T_F is the teaching factor given by Kommadath and Kotecha (2017) as

$$T_F = round(1 + r). \tag{4.7}$$

Then the solution is bounded and its fitness evaluated. If it is better than the solution undergoing the teaching phase, it replaces it (Kommadath; Kotecha, 2017). If it is not better it is discarded and a new solution is generated through the learning phase by

$$X_n^{new} = X_n + r(X_n^{partner1} - X_n) + r(X_n - X_n^{partner2}) \quad \forall n = 1, \dots, N, \tag{4.8}$$

where $X_n^{partner1}$ is the n^{th} design variable of a superior random partner, while $X_n^{partner2}$ is n^{th} design variable from a inferior random partner (Kommadath; Kotecha, 2017). If the solution is better it is accepted, if not it is discarded. The procedure repeats itself until it reaches the stop criteria.

Finally, it should be noted that the mean of the class is updated every time a student completes one of the two phases of the algorithm.

4.3.3 Statistical Benchmark Comparison

The experiments conducted in this section followed the CEC2017 rules. A-GEO$_2$ was run 51 independent times with uniform random initialization for every function, except for B2 that was not computed. For Each function, D was set to 10 and 30, A-GEO$_2$ variables used a 16 bits encoding and the stop criteria was NFE$\geq$100,000.

Table 4.15 and Table 4.16 show the results of A-GEO$_2$ and TLBO-FL for $D = 10$ and $D = 30$, where the columns are the average error between the best, worst, median and mean results achieved by the algorithms and the global optimum, and the standard deviation.

As the results observed show, A-GEO$_2$ performed poorly on functions B1, B3, B12, B13, B14, B15, B18 and B19. These functions are the unimodal and hybrid functions. The awful performance on B12, B13, B15 and B19 can be explained by the composition of these hybrid functions, which includes B1, the worst performance of A-GEO$_2$. Since A-GEO$_2$ presented difficulties handling unimodal function it also can be assumed this is the reason behind the poor performance on B14 and B18.

Considering the average mean error, A-GEO$_2$ only beats TBOL-FL in function B10, a multimodal function with a deceiving global optimal. Also, for the rest of the multimodal and composite functions A-GEO$_2$ performance is competitive with TBOL-FL.

Next a non parametric pairwise sign test (SHESKIN, 2006) comparison was made between both algorithms to identify which one performed better on this test suite statistically.

The signed test is one of the most simple non parametric statistical methods. It was explained by Derrac et al. (2011) as a comparison method that counts how many times an algorithm won over other in a overall comparison. So, to satisfy the null hypotheses of both algorithms performing equally, each algorithm would have to win approximately $F/2$ out of F problems (DERRAC et al., 2011). Thus, to reject the null hypotheses using a confidence interval of 95% ($\alpha = 0.05$), the number of wins, $wins$, of one algorithm should be at least equals to:

$$wins = F/2 + 1.96\sqrt{F}/2. \tag{4.9}$$

Applying the sign test to the average mean error data shown in Table 4.15 and Table 4.16 it is observed that A-GEO$_2$ has only one win in each scenario against TLBO-FL. Since the minimum number of victories for one algorithm to be considered better than other in one scenario is close to 20 for this case (see Equation (4.9)), the null hypothesis can be rejected and there is a strong indication that the later algorithm performed better on this test suite than A-GEO$_2$.

Table 4.15 - Error values of A-GEO$_2$ and TLBO-FL on the 29 benchmark functions for $D = 10$.

	A-GEO$_2$					TLBO-FL				
No.	Best	Worst	Median	Mean	Std. Dev	Best	Worst	Median	Mean	Std. Dev.
B1	5.97E+06	2.87E+09	9.50E+08	1.08E+09	8.20E+08	**3.30E+00**	**9.80E+03**	**1.30E+03**	**2.00E+03**	**2.50E+03**
B2	-	-	-	-	-	-	-	-	-	-
B3	6.48E+03	4.93E+04	2.53E+04	2.58E+04	9.99E+03	**1.30E-11**	**5.60E-03**	**1.50E-08**	**1.10E-04**	**7.80E-04**
B4	5.05E+00	1.27E+02	3.52E+01	4.55E+01	3.17E+01	**1.50E-01**	**4.70E+00**	**3.30E+00**	**3.00E+00**	**1.20E+00**
B5	9.12E+00	7.13E+01	2.38E+01	2.52E+01	1.09E+01	**2.00E+00**	**2.10E+01**	**7.20E+00**	**8.80E+00**	**5.60E+00**
B6	4.84E-01	2.32E+01	4.75E+00	6.08E+00	4.85E+00	**0.00E+00**	**2.90E-06**	**0.00E+00**	**8.40E-08**	**4.40E-07**
B7	2.24E+01	1.07E+02	5.68E+01	5.69E+01	1.75E+01	**2.00E+01**	**4.00E+01**	**2.70E+01**	**2.80E+01**	**4.00E+00**
B8	1.03E+01	4.69E+01	2.66E+01	2.74E+01	9.98E+00	**2.20E+00**	**2.10E+01**	**1.20E+01**	**1.20E+01**	**4.40E+00**
B9	4.79E+00	5.42E+02	9.93E+01	1.39E+02	1.24E+02	**0.00E+00**	**4.50E-01**	**0.00E+00**	**8.90E-03**	**6.40E-02**
B10	2.62E+02	**1.26E+03**	**7.15E+02**	**6.98E+02**	1.88E+02	**3.30E+02**	1.40E+03	9.90E+02	9.50E+02	**2.10E+02**
B11	2.02E+01	2.38E+04	4.41E+03	3.95E+03	5.54E+03	**3.60E-01**	**7.70E+00**	**4.30E+00**	**4.10E+00**	**1.50E+00**
B12	5.42E+03	3.87E+08	1.22E+06	1.57E+07	5.54E+07	**7.40E+03**	**3.00E+05**	**5.90E+04**	**6.60E+04**	**5.50E+04**
B13	**2.17E+01**	5.86E+08	1.70E+07	4.81E+07	9.29E+07	2.00E+02	**1.00E+04**	**1.80E+03**	**2.40E+03**	**2.20E+03**
B14	**2.34E+01**	3.39E+08	3.44E+03	2.66E+07	9.21E+07	3.50E+01	**1.30E+02**	**6.50E+01**	**6.70E+01**	**1.80E+01**
B15	**1.60E+01**	3.36E+07	2.15E+03	1.32E+06	6.59E+06	5.20E+01	**2.50E+02**	**1.20E+02**	**1.30E+02**	**4.30E+01**
B16	1.06E+01	6.47E+02	2.63E+02	2.70E+02	1.58E+02	**1.40E+00**	**1.20E+02**	**3.80E+00**	**8.90E+00**	**2.20E+01**
B17	**1.15E+01**	3.81E+02	8.59E+01	1.05E+02	8.21E+01	2.50E+01	**5.30E+01**	**3.80E+01**	**3.80E+01**	**7.80E+00**
B18	**7.47E+01**	4.93E+06	6.61E+03	2.35E+05	9.64E+05	3.80E+02	**2.60E+04**	**4.30E+03**	**6.20E+03**	**5.60E+03**

(Continue)

Table 4.15 - Conclusion

	A-GEO$_2$					TLBO-FL				
No.	Best	Worst	Median	Mean	Std. Dev	Best	Worst	Median	Mean	Std. Dev.
B19	**9.70E+00**	1.39E+09	2.04E+03	2.73E+07	1.95E+08	1.50E+01	**1.50E+02**	**5.30E+01**	**6.10E+01**	**3.20E+01**
B20	6.02E+00	3.84E+02	9.00E+01	1.09E+02	8.43E+01	**1.00E+00**	**2.60E+01**	**2.10E+01**	**1.50E+01**	**9.40E+00**
B21	1.08E+02	2.56E+02	2.27E+02	2.17E+02	**3.28E+01**	**1.00E+02**	**2.20E+02**	**1.10E+02**	**1.40E+02**	5.20E+01
B22	1.24E+02	1.39E+03	7.56E+02	6.75E+02	4.19E+02	**1.20E+01**	**1.00E+02**	**1.00E+02**	**9.30E+01**	**2.30E+01**
B23	3.06E+02	4.28E+02	3.27E+02	3.30E+02	1.71E+01	**3.00E+02**	**3.20E+02**	**3.10E+02**	**3.10E+02**	**3.80E+00**
B24	1.88E+02	3.87E+02	3.56E+02	3.25E+02	**6.60E+01**	**1.00E+02**	**3.40E+02**	**3.30E+02**	**3.10E+02**	6.90E+01
B25	**2.32E+02**	8.22E+02	4.59E+02	4.81E+02	9.74E+01	4.00E+02	**4.50E+02**	**4.40E+02**	**4.30E+02**	**2.20E+01**
B26	2.65E+02	1.57E+03	5.94E+02	8.76E+02	4.64E+02	**0.00E+00**	**3.60E+02**	**3.00E+02**	**3.00E+02**	**4.60E+01**
B27	3.91E+02	4.96E+02	4.14E+02	4.25E+02	2.73E+01	**3.90E+02**	**4.00E+02**	**3.90E+02**	**3.90E+02**	**3.30E+00**
B28	3.74E+02	9.43E+02	6.13E+02	5.77E+02	**1.39E+02**	**3.00E+02**	**9.30E+02**	**3.70E+02**	**4.50E+02**	1.60E+02
B29	2.51E+02	4.69E+02	3.33E+02	3.45E+02	5.82E+01	**2.50E+02**	**3.00E+02**	**2.70E+02**	**2.70E+02**	**1.40E+01**
B30	1.33E+03	2.36E+06	1.91E+05	5.93E+05	7.19E+05	**1.10E+03**	**1.30E+06**	**9.10E+03**	**2.80E+05**	**4.90E+05**

Fonte: Adapted by the author from Kommadath and Kotecha (2017).

Table 4.16 - Error values of A-GEO$_2$ and TLBO-FL on the 29 benchmark functions for $D = 30$.

	A-GEO$_2$					TLBO-FL				
No.	Best	Worst	Median	Mean	Std. Dev	Best	Worst	Median	Mean	Std. Dev.
B1	4.72E+08	8.40E+09	3.91E+09	4.22E+09	1.91E+09	**6.20E-03**	**1.60E+04**	**2.30E+03**	**3.50E+03**	**3.60E+03**
B2	-	-	-	-	-	-	-	-	-	-
B3	9.75E+04	2.87E+05	1.96E+05	1.95E+05	4.75E+04	**9.10E+02**	**5.80E+03**	**2.90E+03**	**3.00E+03**	**1.10E+03**
B4	1.33E+02	1.55E+03	3.33E+02	3.92E+02	2.36E+02	**2.90E+01**	**1.40E+02**	**8.70E+01**	**9.00E+01**	**2.40E+01**
B5	8.69E+01	2.12E+02	1.53E+02	1.52E+02	2.97E+01	**1.80E+01**	**1.40E+02**	**3.60E+01**	**4.00E+01**	**2.10E+01**
B6	3.32E+00	1.92E+01	1.02E+01	1.00E+01	3.85E+00	**1.70E-02**	**1.80E+00**	**3.90E-01**	**4.90E-01**	**4.20E-01**
B7	1.73E+02	3.93E+02	2.73E+02	2.71E+02	5.52E+01	**5.30E+01**	**2.10E+02**	**1.50E+02**	**1.40E+02**	**4.70E+01**
B8	7.74E+01	2.28E+02	1.46E+02	1.47E+02	3.65E+01	**1.80E+01**	**1.20E+02**	**3.10E+01**	**3.70E+01**	**1.80E+01**
B9	1.04E+03	1.41E+04	5.53E+03	5.79E+03	2.49E+03	**2.40E+00**	**1.30E+02**	**3.00E+01**	**3.40E+01**	**2.70E+01**
B10	**2.59E+03**	**4.48E+03**	**3.54E+03**	**3.49E+03**	4.81E+02	6.00E+03	7.20E+03	6.70E+03	6.70E+03	**2.80E+02**
B11	1.56E+03	3.24E+04	1.08E+04	1.19E+04	7.52E+03	**2.60E+01**	**2.00E+02**	**8.20E+01**	**8.20E+01**	**4.10E+01**
B12	1.76E+07	2.35E+09	2.90E+08	4.61E+08	5.31E+08	**5.60E+03**	**5.40E+05**	**3.60E+04**	**5.70E+04**	**9.00E+04**
B13	6.41E+07	1.11E+10	7.02E+08	1.33E+09	1.88E+09	**2.50E+02**	**8.60E+04**	**1.60E+04**	**2.00E+04**	**1.80E+04**
B14	1.22E+05	7.62E+07	6.03E+06	1.22E+07	1.60E+07	**7.40E+02**	**3.20E+04**	**6.00E+03**	**7.10E+03**	**5.80E+03**
B15	3.44E+03	2.77E+09	1.61E+08	4.71E+08	7.07E+08	**2.10E+03**	**1.30E+05**	**1.30E+04**	**2.20E+04**	**2.30E+04**
B16	8.04E+02	1.91E+03	1.37E+03	1.36E+03	2.83E+02	**8.90E+00**	**1.50E+03**	**3.80E+02**	**4.90E+02**	**3.50E+02**
B17	3.42E+02	1.28E+03	7.63E+02	7.81E+02	2.30E+02	**4.50E+01**	**3.00E+02**	**1.20E+02**	**1.40E+02**	**6.60E+01**
B18	3.25E+05	7.41E+07	9.43E+06	1.47E+07	1.82E+07	**9.00E+04**	**8.70E+05**	**3.40E+05**	**3.70E+05**	**1.70E+05**

(Continue)

Table 4.16 - Conclusion

	A-GEO$_2$					TLBO-FL				
No.	Best	Worst	Median	Mean	Std. Dev	Best	Worst	Median	Mean	Std. Dev.
B19	7.11E+04	6.08E+09	2.88E+08	6.89E+08	1.23E+09	**6.60E+02**	**5.50E+04**	**7.10E+03**	**1.10E+04**	**1.10E+04**
B20	2.19E+02	1.49E+03	7.11E+02	7.21E+02	2.27E+02	**4.90E+01**	**5.40E+02**	**2.10E+02**	**2.20E+02**	**1.20E+02**
B21	2.46E+02	4.19E+02	3.51E+02	3.49E+02	3.44E+01	**2.20E+02**	**2.60E+02**	**2.30E+02**	**2.30E+02**	**1.20E+01**
B22	3.57E+02	5.48E+03	4.03E+03	3.99E+03	9.02E+02	**1.00E+02**	**1.10E+02**	**1.00E+02**	**1.00E+02**	**1.90E+00**
B23	4.00E+02	5.94E+02	5.11E+02	5.10E+02	4.40E+01	**3.70E+02**	**4.30E+02**	**3.90E+02**	**4.00E+02**	**1.60E+01**
B24	5.69E+02	9.25E+02	7.09E+02	7.10E+02	8.49E+01	**4.50E+02**	**5.10E+02**	**4.70E+02**	**4.70E+02**	**1.60E+01**
B25	4.18E+02	1.37E+03	6.84E+02	7.23E+02	2.12E+02	**3.80E+02**	**4.40E+02**	**3.90E+02**	**4.00E+02**	**1.80E+01**
B26	8.15E+02	5.49E+03	2.87E+03	2.77E+03	8.25E+02	**2.00E+02**	**2.20E+03**	**1.50E+03**	**1.40E+03**	**4.70E+02**
B27	5.19E+02	6.71E+02	5.76E+02	5.75E+02	3.25E+01	**5.00E+02**	**5.90E+02**	**5.30E+02**	**5.30E+02**	**2.10E+01**
B28	5.48E+02	1.81E+03	8.44E+02	9.09E+02	2.54E+02	**3.80E+02**	**4.80E+02**	**4.30E+02**	**4.30E+02**	**2.70E+01**
B29	5.51E+02	1.72E+03	1.09E+03	1.11E+03	2.53E+02	**5.00E+02**	**8.00E+02**	**5.90E+02**	**6.20E+02**	**9.10E+01**
B30	6.27E+04	4.84E+08	2.38E+07	6.65E+07	1.11E+08	**4.10E+03**	**1.70E+05**	**1.90E+04**	**2.60E+04**	**2.80E+04**

Fonte: Adapted by the author from Kommadath and Kotecha (2017).

In summary, A-GEO$_2$ has to be improved to compete with the state of the art evolutionary algorithms. Considering that A-GEO$_2$ is still the first adaptive version of GEO and GEO is almost 20 years old, the performance results were reasonable. A-GEO$_2$ had difficulties with unimodal functions which undermined its performance on the hybrid functions that used them, however for the rest of the functions presented it was competitive with TBOL-FL and even won in one case. Also, A-GEO$_2$ is a parameterless version of GEO which gives it another huge advantage that is not computed on this test, since TBOL-FL has the population size as a parameter that needs to be **tunned**.

For last, one of the possible reasons for the gap in performance can be the variables encoding used, since A-GEO$_2$ still uses binary encoding while the more recent algorithms use real values. However, further analysis and improvements need to be done to A-GEO to understand why it had such a disadvantage on unimodal functions.

5 SPACECRAFT CONCEPT DESIGN APPLICATION

For Earth observation satellites in Low Earth Orbits (LEOs), the design of spacecraft is driven by payload parameters such as mass and power (WERTZ et al., 2011). It is known that the lower the orbit, the simpler the camera design, due to the lower distance between the equipment and the target. However, a closer orbit will place the satellite on an environment with higher air density and, consequently, higher drag force, requiring a huge amount of fuel to maintain the nominal orbit (CHAGAS et al., a). These aforementioned trade-offs exemplify the complexity of optimization design in the conceptual phase, which encourages the use of MDO techniques.

Extending the works of Chagas et al. (b) and Chagas et al. (a), a simplified case study is proposed in which the algorithms described in the Chapter 3 are used as optimizers in an MDO to solve the following problem of the conceptual space mission design phase:

- **Find the variables I, Q and D, which define a Sun-synchronous ground repeating orbit (SSGRO) (CHAGAS et al., a; WERTZ et al., 2011), and minimize the satellite total mass S_{m} composed of the satellite dry mass m_{d} and propellant mass m_{p}.**

The design variables required (I, Q and D) define the number of orbit revolutions that a satellite makes per day (rev), given by

$$\text{rev} = I + \frac{Q}{D} \qquad I, Q \in \mathbb{Z}_{+}, \quad D \in \mathbb{Z}_{+}^{*}, \tag{5.1}$$

where I, Q, D are integer numbers in which $Q < D$. Notice that after D days, the satellite will complete $D \cdot I + Q$ revolutions, which is an integer number, leading to the ground-track repetition in D days.

This chapter describes, discusses and evaluates the developed algorithms efficiency applied into an MDO to solve a spacecraft conceptual design optimization case study. First the mathematical models used for each necessary discipline are explained in detail, then the problem mathematical definition is presented, followed by the MDO framework and architecture. At last, the results are presented and analyzed.

5.1 Orbit analysis model

The orbit analysis model finds a nominal orbit for the mission and computes the total velocity variation ΔV_{t} budget that the satellite propulsion subsystem shall be capable to provide. Where the ΔV_t budget is the sum of the amount of change in the orbital velocity that the spacecraft must apply to transfer itself from the injection orbit to the nominal orbit, to maintain its orbit altitude during the mission lifetime, and transfer itself to the disposal orbit (CHAGAS et al., a).

5.1.1 Nominal orbit

The Sun-synchronous orbits (SSOs) are the most used ones for Earth observation missions. They provide global coverage and the lightning conditions on the ground are kept almost constant during the mission lifetime, which is important when comparing captured images (VALLADO, 2007). A SSO is an orbit in which the gravitational perturbation precesses the right ascension of the ascending node (RAAN) by 0.9856473598948°/day (WERTZ et al., 2011). In this case, the orbit plane will ideally have the same orientation with respect to the Sun during the entire year. The RAAN time derivative considering perturbation terms up to the Earth second gravitational zonal harmonic $J2$ is given by (WERTZ et al., 2011)

$$\dot{\Omega} = -\frac{3}{2}\frac{R_0^2}{a^2(1-e^2)^2}n_0 J_2 \mathrm{cos}i, \tag{5.2}$$

where $R_0 = 6,378,137$ m is the Earth equatorial radius, a is the semi-major axis, e is the eccentricity, $J_2 = 1.08262617385222 \times 10^{-3}$ is the Earth second gravitational zonal harmonic, i is the orbit inclination, and n_0 is the the non-perturbed orbit angular velocity, given by

$$n_0 = \sqrt{\frac{\mu_0}{a^3}}, \tag{5.3}$$

where $\mu_0 = 3.986004418 \times 10^{14}$ m^3/s^2 is Earth standard gravitational parameter.

Earth observation missions usually requires for the satellite to scan multiple times the same area during a period, known as revisit period. This is accomplished by selecting the number of orbital revolutions per day (rev), as in Equation (5.1) (VALLADO, 2007). Hence, the camera can be designed so that an image of a target can be obtained periodically according to this revisit time. Thus, solving

$$\begin{cases} 1.99106385344371 97 \times 10^{-7} - \dot{\Omega} = 0 \\ n_i - n_0 - \frac{3}{4}\frac{R_0^2 J_2}{a^2(1-e^2)^2} n_0 \{\sqrt{1-e^2}[3\cos^2(i) - 1] + 5\mathrm{cos}^2 i - 1\} = 0, \end{cases} \tag{5.4}$$

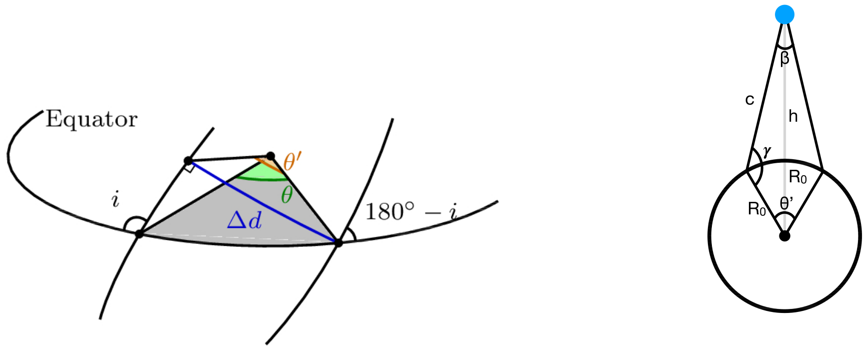
for the semi-major axis a and the inclination i results in an SSGRO, whereas the orbital angular velocity n_i is given by

$$n_i = \text{rev}\frac{2\pi}{86400}. \tag{5.5}$$

5.1.1.1 Field of View

After the orbit design, the next step is to find the minimum field of view that the camera must have so that it can obtain images of the entire world in the selected orbit. The minimum FOV is the angular distance, as seen by the satellite, between two adjacent ground tracks (passages) on the Equator. If the camera design meets this constraint, then the instrument will be able to acquire images of any place in the world at every D days. Figure 5.2(a) shows the geometry of the problem where θ' is the angle between two adjacent passages measured from the Earth center and considering the orbit inclination. In this case, the minimum swath width of the optical instrument, which is the length of the imaged area on the ground, is the distance Δd .

Figure 5.1 - Geometries of the orbit and the field of view of the satellite.



(a) Geometry of the distance between two adjacent ground tracks.

(b) Geometry for calculation of the minimum FOV.

SOURCE: Adapted by the author from Chagas and Lopes (2015).

The Earth rotation angle during the orbit period T_i is

$$\Theta = T_i\frac{2\pi}{86400} = \frac{2\pi}{n_i}\frac{2\pi}{86400}. \tag{5.6}$$

Since the orbit cycle is D days, then the satellite will cross the equator $D-1$ times between every two consecutive passages during the orbit period (WERTZ et al., 2011). Hence, the angle θ can be computed as in

$$\theta = \frac{\Theta}{D}, \tag{5.7}$$

and, using spherical trigonometry, one can see that

$$\theta' = \sin^{-1}(\sin(\theta)\sin i). \tag{5.8}$$

Looking from the satellite point of view, shown in Figure 5.2(b), where β is the FOV of the instrument, it follows that the minimum field of view FOV_{min} can be computed by

$$c = \sqrt{R_0^2 + (R_0 + h)^2 - 2R_0(R_0 + h)\text{cos}\frac{\theta'}{2}} \tag{5.9a}$$

$$\beta = 2\text{sin}^{-1}(\frac{R_0}{c}\text{sin}\frac{\theta'}{2}) \tag{5.9b}$$

$$FOV_{\text{min}} = 2\text{sin}^{-1}\left(\frac{R_0}{\sqrt{R_0^2 + (R_0 + h_{\text{p}})^2 - 2R_0(R_0 + h_{\text{p}})\text{cos}\frac{\theta'}{2}}}\text{sin}\frac{\theta'}{2}\right) \tag{5.9c}$$

where h is the satellite altitude. If the selected orbit is eccentric, then one must use the orbit perigee altitude h_{p} to account for the worst-case scenario.

5.1.2 Orbit acquisition

After a spacecraft is launched it arrives at an injection orbit, and later corrective maneuvers are made to position it into its nominal orbit.

The maneuvers correct the spacecraft orbit by adjusting the semi-major axis and the orbit inclination. To correct the semi-major axis of the orbit it is necessary to make two burns of propellant, one to correct its apogee and other to correct its perigee (WERTZ et al., 2011). The first burn is executed at the angular position of the nominal orbit perigee so that the orbit apogee is placed on the desired location. This puts the satellite into a transfer orbit, which is an intermediate elliptical orbit used to transfer a satellite from one orbit to another (WERTZ et al., 2011). The second burn shall be executed in the orbit apogee and places the perigee in the desired position.

Consider the velocity of the spacecraft represented as in Wertz et al. (2011) by

$$V_{a,r} = \sqrt{\mu_0\left(\frac{2}{r} - \frac{1}{a}\right)}, \tag{5.10}$$

where μ is the Earth gravitational constant, a is the semi-major axis of the orbit, and r is the norm of the satellite position vector. As previously stated, adjusting the spacecraft injection orbit semi-major axis to the nominal orbit semi-major axis requires two burns (WERTZ et al., 2011), given by

$$\Delta V_1 = V_{\mathrm{a_t},\mathrm{r_i}} - V_{\mathrm{r_i},\mathrm{r_i}}, \tag{5.11}$$

$$\Delta V_2 = V_{\mathrm{a_n},\mathrm{r_{p_n}}} - V_{\mathrm{a_t},\mathrm{r_{at}}}, \tag{5.12}$$

where a_{n} is the semi-major axis of the nominal orbit, $r_{\mathrm{p_n}}$ is the perigee of the nominal orbit, r_{at} is the apogee of the injection orbit, r_{i} is the injection orbit radius and a_{t} is the semi-major axis of the transfer orbit given by Wertz et al. (2011) as

$$a_t = \frac{r_{\mathrm{p_n}} + r_{\mathrm{i}}}{2}. \tag{5.13}$$

The inclination correction maneuver requires a single burn normal to the orbital plane at the ascending or descending node (WERTZ et al., 2011), it can be calculated by

$$\Delta V_3 = 2V_{\mathrm{a_n},\mathrm{r_{an}}} \sin\left(\frac{\Delta i}{2}\right), \tag{5.14}$$

where Δi is the inclination error and r_{an} is the nominal orbit apogee.

Finally, to compute the total velocity needed to correct the satellite orbit, the sum of the the velocity variations modules obtained is made by

$$\Delta V_{\mathrm{oa}} = |\Delta V_1| + |\Delta V_2| + |\Delta V_3|. \tag{5.15}$$

5.1.3 Orbit maintenance

Since for this case study only LEO orbits are being considered, the satellite should be able to correct the displacement caused by the atmospheric drag to sustain itself through its mission lifetime in the nominal orbit as explained by Chagas et al. (a, p. 3):

> Remote sensing mission with optical payloads are placed on a Low Earth Orbit (LEO), which has altitudes from 400 km up to 1200 km (Wertz et al., 2011). In these orbits, the atmospheric density applies a drag force to the satellite that removes energy from the orbit and, consequently, decreases the orbit altitude (Vallado and McClain, 2004).

Considering near circular orbits, the velocity variation ΔV_{om} necessary to counterbalance the drag force can be computed as in Chagas et al. (a):

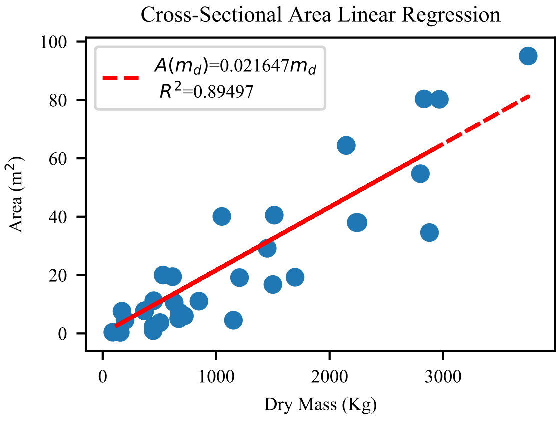
$$\Delta V_{\mathrm{om}} = \pi \frac{C_{\mathrm{D}} A_{\mathrm{cs}}}{S_{\mathrm{m}}} \rho a_{\mathrm{n}} V_{\mathrm{a_n}, \mathrm{r_{p_n}}} \frac{T_{\mathrm{m}}}{T_{\mathrm{orb}}}, \tag{5.16}$$

where C_{D} is the drag coefficient, A_{cs} is the satellite cross-sectional area, S_{m} is the satellite total mass, ρ is the atmospheric density, T_{m} is the expected mission lifetime in seconds, and T_{orb} is the orbital period in seconds, given by Wertz et al. (2011) as

$$T_{\mathrm{orb}} = 2\pi \sqrt{\frac{r_{\mathrm{p_n}}}{\mu_0}}. \tag{5.17}$$

The satellite cross-sectional area A_{cs} is estimated by a parametric model developed taking into account a relation between itself and the satellite dry mass. Data from 34 LEO satellites, shown in Figure 5.2, launched between years 2000 and 2017, that have an Earth Observation device were collected from the UCS Satellite Database.

Figure 5.2 - Satellite's cross-sectional area and dry mass relation.



SOURCE: Made by the author.

Computing the satellite cross-section area from the data collected (A_{cs0}) required to draw a plane cutting the spacecraft perpendicular to its orbit, and if the solar panels were not mounted on the body of the satellite, their area was take into account. Thus, it was computed as:

$$A_{\mathrm{cs0}} = \begin{cases} A_{\mathrm{sp}} + A_{\mathrm{p}} & \text{if the solar panel is deployable,} \\ A_{\mathrm{p}} & \text{otherwise,} \end{cases} \tag{5.18}$$

where A_{sp} is the solar panel area and A_{p} is the satellite platform cross-section area. Then making a linear regression of the data collected, between dry mass and cross-sectional area, it is concluded that the satellite cross sectional area A_{cs} is given by

$$A_{\mathrm{cs}}(m_{\mathrm{d}}) = 0.0216m_{\mathrm{d}}, \tag{5.19}$$

where m_{d} is the satellite dry mass. It is important to notice that in Chagas et al. (a) this relation was found to be $0.019646m_{\mathrm{d}}$, which is not far from the one found on on this work.

5.1.4 Deorbit

A disposal orbit is an orbit that will dispose the spacecraft (i.e atmospheric reentering) after a specific period. This implies a propellant consumption at the end of the satellite lifetime to insert it into this orbit, if so is required.

Nowadays exists an agreement between space agencies to dispose their satellites after 25 years of their end of operation. Thus, a model to estimate a spacecraft disposal orbit that fulfills this agreement was developed, where the perigee of the nominal orbit is changed to a perigee that turns it into a disposal orbit.

Using Debris Assessment Software (DAS) v2.0.2 it was possible to calculate the orbit cycles of a satellite given its ballistic coefficient S_{bc}, apogee altitude h_{a}, perigee altitude h_{p} and orbit inclination i during a specific period, thus showing when the satellite will reenter atmosphere.

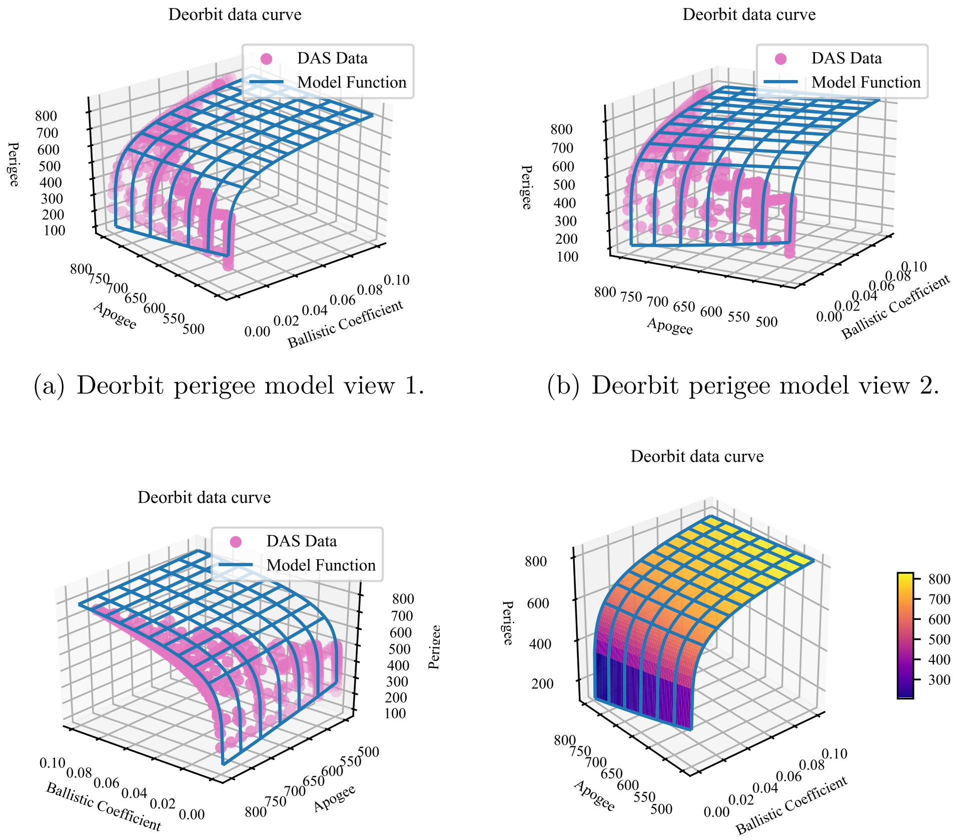
The software was used to create a model to find a perigee altitude h_{p} that predicted atmosphere reenter in a lifespan of 25 years by randomly generating 360 points of apogee altitude h_{a} and satellite ballistic coefficients S_{bc} between 500 and 800 km and 0.0001 and 0.9 kg/m^2 respectively, with a nominal orbit inclination i of 98 degrees. Also, using as starting year 2017, since it was the worst case scenario due to the year solar cycle.

After all points were generated, the 3D model was plotted and it is shown in Figure 5.3. Also, using GNU OCTAVE v4.2.1 to interpolate the points, one gets:

$$h_{\mathrm{p}} = -2.96741 \ln(S_{\mathrm{bc}}).[38.8249 \exp(-0.001 h_{\mathrm{a}}) - 50.7627] + 1017.1592, \qquad (5.20)$$

where h_{a} is the apogee altitude and h_{p} the perigee altitude.

Figure 5.3 - DAS output vs Model.



(a) Deorbit perigee model view 1.

(b) Deorbit perigee model view 2.

(c) Deorbit perigee model view 3.

(d) Deorbit perigee model view 4.

SOURCE: Made by the author.

The Equation (5.20) was compared to the other 360 points of Perigee altitude collected from DAS and the mean error of this model is 0.0019Km with a standard deviation of 18.9913 Km.

Now, having the disposal orbit perigee it is possible to compute the change in velocity ΔV_{od}, given as

$$\Delta V_{\mathrm{od}} = V_{\mathrm{an,r_{an}}} - V_{\mathrm{ad,r_{ad}}}, \qquad (5.21)$$

where a_{d} is the disposal orbit semi-major axis, and r_{ad} is the disposal orbit apogee. The disposal orbit semi-major axis a_{d} can be computed as

$$a_{\mathrm{d}} = \frac{r_{\mathrm{ad}} + r_{\mathrm{pd}}}{2}, \tag{5.22}$$

where r_{pd} is the disposal orbit perigee.

5.2 Payload model

To estimate the instrument mass, power, and FOV for the desired mission, a model described in Wertz et al. (2011) was used. This method scales a new instrument from a referenced one based on the instrument optical aperture ratio R defined by

$$R = \frac{A}{A_0}, \tag{5.23}$$

where A_0 is the aperture of the reference instrument. The desired instrument optical aperture A is a function of the orbit height h, and focal length f as described in Chagas et al. (a) as

$$f = \frac{P_{\mathrm{size}} h}{Res_{\mathrm{N}}}, \tag{5.24a}$$

$$A = A_0 R_{\mathrm{v}} \frac{f}{f_0}, \tag{5.24b}$$

where R_{v} is the ratio between the ground velocity at the Equator for the current mission and the payload reference mission, both measured at the descendant node, P_{size} is the instrument pixel size, and Res_{N} is the desired mission resolution.

Afterwards computing the ratio R_{v}, the mission payload head mass W_{opt} and power P_{p} are estimated as in (WERTZ et al., 2011) by

$$W_{\mathrm{opt}} = K R^3 W_{\mathrm{opt0}}, \tag{5.25a}$$

$$P_{\mathrm{p}} = K R^3 P_{\mathrm{p0}}, \tag{5.25b}$$

where W_{opt0} is the optical head mass of the reference instrument, P_{p0} is the power of the reference instrument and K is a scaling factor given by

$$K = \begin{cases} 2 & forR < 0.5, \\ 1 & otherwise. \end{cases} \tag{5.26}$$

Considering the payload mass W_{p} as the sum of the optical head mass W_{opt} and its electronics mass W_e one gets:

$$W_\mathrm{p} = W_\mathrm{opt} + W_\mathrm{e}. \tag{5.27}$$

Finally, the payload field of view FOV_payload can be computed by

$$FOV_\mathrm{payload} = 2 \arctan \frac{N_\mathrm{px} P_\mathrm{size}}{2f}, \tag{5.28}$$

where N_px is the number of pixels of the camera.

5.3 Propulsion model

The total velocity variation for the entire mission lifetime ΔV_t is given by

$$\Delta V_\mathrm{t} = |\Delta V_\mathrm{oa}| + |\Delta V_\mathrm{om}| + |\Delta V_\mathrm{od}|. \tag{5.29}$$

Given the total velocity ΔV_t and the satellite total mass S_m, the propellant mass m_p can be obtained as in Wertz et al. (2011) by

$$m_\mathrm{p} = S_\mathrm{m}\left(1 - e^{-\frac{\Delta V}{g_0 I_\mathrm{sp}}}\right) \tag{5.30}$$

where $g_0 = 9.80665$ m/s is the gravity acceleration at Earth's surface and I_sp is the propellant specific impulse, which is 225s for hydrazine (WERTZ et al., 2011).

5.4 Satellite model

The estimation of the total satellite mass requires a recursive algorithm since it is necessary to know a priori the total satellite mass S_m to compute the satellite propellant mass m_p, and the contrary is also true. Developed by Chagas et al. (a), the algorithm is shown below:

1. Given the orbit and a reference instrument, obtain the payload total mass W_p as in Section Section 5.2;

2. Obtain the satellite dry mass estimate using the parametric relation (WERTZ et al., 2011)

$$m_\mathrm{d} = W_\mathrm{p}/0.31; \tag{5.31}$$

3. Obtain the satellite total mass estimate using the parametric relation of

the first iteration, (WERTZ et al., 2011)

$$\underset{0}{S_{\mathrm{m}}} = 1.27m_{\mathrm{d}}; \tag{5.32}$$

4. Compute the satellite cross-sectional area using the model described on Section Subsection 5.1.3 using the dry mass m_{d};

5. FOR $j = 1 : max$ DO

    5.1. Compute the total velocity variation ΔV_{t} using the orbit analysis model described in Section Section 5.1 using the total mass $\underset{j-1}{S_{\mathrm{m}}}$;

    5.2. Compute the fuel mass of the j^{th} iteration $\underset{j}{m_{\mathrm{p}}}$ using the propulsion model described in Section Section 5.3;

    5.3. Compute the new estimate for the satellite total mass:

    $$\underset{j}{S_{\mathrm{m}}} = m_{\mathrm{d}} + \underset{j}{m_{\mathrm{p}}} \tag{5.33}$$

    5.4. If $|\underset{j}{S_{\mathrm{m}}} - \underset{j-1}{S_{\mathrm{m}}}| < \epsilon$, stop.

6. END FOR

The threshold ϵ has been selected as 1E+0−5 kg and max has been selected as 100. It has been verified that if the orbit altitude is higher than 360 km, the algorithm converges. Otherwise, the stop condition is never reached and after each iteration the satellite total mass estimate is increased until $j = max$ (CHAGAS et al., a).

Next, for the satellite power P, Wertz et al. (2011) gives the following relation

$$P = P_p/0.46. \tag{5.34}$$

Now that all the necessary disciplines models were explained, the MDO framework implemented is presented in the next section.

5.5 Multidisciplinary Optimization Framework

The spacecraft optimization model can be put as:

$$\begin{aligned}
&\underset{S_\mathrm{m}(I,N,D)}{\text{Minimize}} && S_\mathrm{m}(I,Q,D) \\
&\text{Where} && S_\mathrm{m} = m_\mathrm{d} + m_\mathrm{p} \\
\\
&\text{Subject}\,to && FOV_\mathrm{payload} - 1.05FOV_\mathrm{min} \geq 0 \\
& && I^L \leq I \leq I^U \\
& && 1 \leq D \leq D^U \\
& && 1 \leq Q < D
\end{aligned} \tag{5.35}$$

where m_d (Equation (5.31)) is the satellite dry mass, m_p (Equation (5.30)) is the satellite propellant mass, FOV_payload is the payload field of view (Equation (5.28)), FOV_min is the minimum payload FOV in the nominal orbit (Equation (5.9c)) and I^L, I^U, and D^U are the lower and upper boundaries of the respective design variables.

Notice that to ensure that the satellite payload can record images of the entire Earth surface it is necessary that the payload instrument field of view, FOV_payload, be greater or equal to the minimum payload field of view in the nominal orbit, FOV_min, plus a margin.

After the problem definition is established the MDO requires to be assembled into one architecture that will integrate its optimization model, design variables and optimizers. In respect to the MDO architecture adopted, it was the Multidisciplinary Feasible (MDF) architecture. It was chosen due to its advantages of always returning a system design that respects the optimization problem constrains and let the optimizer have under direct control only the design variables, objective function and constraints (MARTINS; LAMBE, 2013).

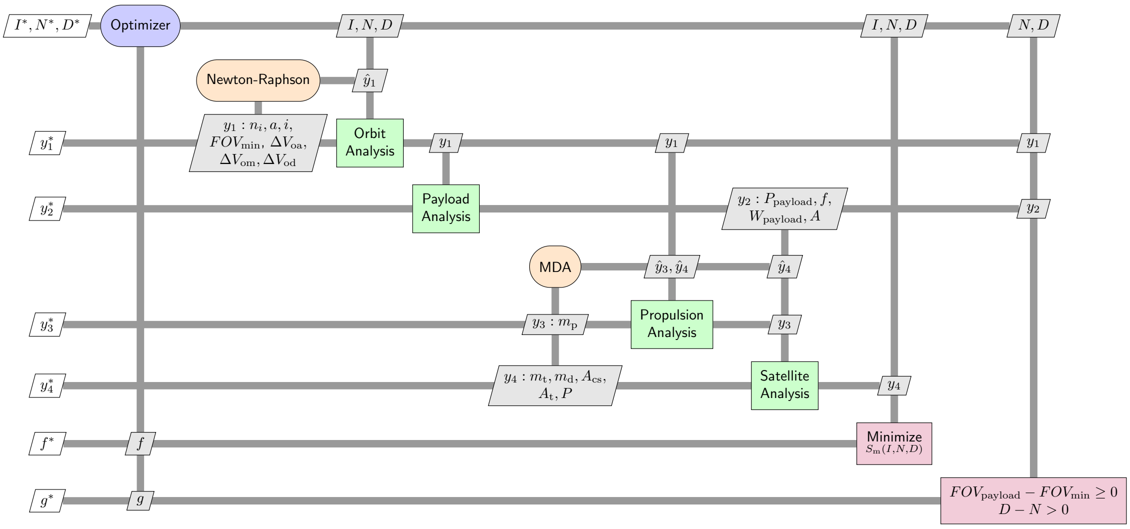
Finally, the complete MDO for this optimization problem is presented in the format of an XDSM[1] in Figure 5.4. Notice that $\hat{y}_\mathrm{a}$ is the copy of the response variables y_a, computed in each analyses a, while y^*_a is the final solution, found in the last iteration.

First, the MDO starts the optimization algorithm that generates a random Sun-Synchronous Ground Repeating Orbit (SSGRO). At each iteration of the optimizer,

[1] For more information about XDSM refer to Lambe and Martins (2012).

the orbit analyses, payload design, satellite design and propulsion analysis models, are evaluated. These models were explained in detail in Section 5.2, Section 5.1, Section 5.3 and Section 5.4.

Figure 5.4 - XDSM for spacecraft conceptual design.


I^*, N^*, D^*
Optimizer
I, N, D
I, N, D
N, D
Newton-Raphson
$\hat{y}_1$
y_1^*
$y_1 : n_i, a, i,$ FOV_{min}, ΔV_{oa}, $\Delta V_{\text{om}}, \Delta V_{\text{od}}$
Orbit Analysis
y_1
y_1
y_1
y_2^*
Payload Analysis
$y_2 : P_{\text{payload}}, f,$ W_{payload}, A
y_2
MDA
$\hat{y}_3, \hat{y}_4$
$\hat{y}_4$
y_3^*
$y_3 : m_{\text{p}}$
Propulsion Analysis
y_3
y_4^*
$y_4 : m_{\text{t}}, m_{\text{d}}, A_{\text{cs}},$ A_{t}, P
Satellite Analysis
y_4
f^*
f
Minimize $S_{\text{m}}(I,N,D)$
g^*
g
$FOV_{\text{payload}} - FOV_{\text{min}} \geq 0$
$D - N > 0$

SOURCE: Made by the author.

After the MDO initialization, first a nominal SSGRO and its minimum field of view (FOV_{min}) are computed based on the orbit revolution, as described in Subsection 5.1.1. Afterwards, based on a reference payload chosen by the user, the mission payload is designed as described in Section 5.2.

After designing the payload for the mission, the satellite design is started. With a nominal orbit, a designed payload and the launcher errors, which is given by the user as input parameters, the satellite design model computes the satellite dry mass m_{d} and power P as in Section 5.4. With the satellite dry mass, its area A_{t} and ballistic coefficient S_{bc} are computed from a parametric linear regression model developed to obtain the satellite cross-sectional area A_{cs} through the satellite dry mass m_{d} using the UCS Satellite Database, described in Subsection 5.1.3.

Having the satellite area, the total propellant mass m_{p} for the orbit acquisition, maintenance and disposal maneuvers is estimated as described in Section 5.1, Section 5.3 and Section 5.4. For the disposal maneuvers fuel consumption computation it is necessary to provide the disposal orbit perigee h_{p}, thus an orbit analyses model was developed and is described in Subsection 5.1.4.

Finally the total satellite mass S_{m} is computed using the model described in Section 5.4. If the MDO reached the stop criteria, the results are returned to the user and the MDO is finalized. Otherwise, a new evaluation is computed from a new point selected by the optimizer.

Next section presents the MDO results, using the developed algorithms as the MDO optimizers.

5.6 Spacecraft concept design MDO Results

The parameters selected for the case study can be seen in Table 5.1. For this case study I was used between 13 and 15 otherwise the selected orbit would not be a LEO, which is required for the Earth observation missions we are studying here. At last, D is used between 1 and 60 because those are the minimum and maximum revisit periods desired.

Table 5.1 - Simulation input settings.

Parameter	Value
I	2 bits encoding in the range [13;15]
D	6 bits encoding in the range [1;60]
Q	6 bits encoding in the range [1;D-1]
I_{sp}	225 s
Total area (A_{t}) percentage of cross-sectional area (A_{cs})[1]	180%
Drag coefficient (C_{d})	2.2
Eccentricity of nominal orbit (e)	0.0
Mission lifetime	4 years
Atmospheric density model[2]	1 year with $F10.7 = 225$ and 3 years with $F10.7 = 175$
Mission resolution (Res_{N})	20 m
Launch error in the semi-major axis	20.000 Km
Launch error in the inclination	0.015°

[1] This is the total area of the satellite with respect to the satellite cross-sectional area, in this case the total area is 1.8 times the cross-sectional area.

[2] The mean atmospheric density was obtained from NRLMSISE-00 model.

SOURCE: Adapted by the author from Chagas et al. (a), Wertz et al. (2011).

Before running the optimization, an extensive search was performed on the problem to find its global optimum, as presented in Table 5.2 and took 11:43 minutes on a Microsoft Surface 3 Pro, which has its specifications shown in Table 5.3.

Table 5.2 - Optimal solution.

Variables	Values
I	14
Q	59
D	60
Semi-major axis of nominal orbit (a)	6944.284 Km
Inclination of nominal orbit (i)	97.656°
FOV$_{min}$	4.469°
ΔV for Orbit Acquisition (ΔV_{oa})	30.768 m/s
ΔV forOrbit Maintenance (ΔV_{om})	217.983 m/s
ΔV for De-orbiting maneuver (ΔV_{od})	0 m/s
Dry mass (m_{d})	175.952 Kg
Cross-sectional area (A_{cs})	3.801 m^2
Total area (A_{t})	6.841 m^2
Propellant mass (m_{p})	20.997 Kg
Total mass (m_{t})	196.949 Kg
Power (P)	114.769 W

SOURCE: Made by the author.

Table 5.3 - Microsoft Surface 3 Pro Specifications.

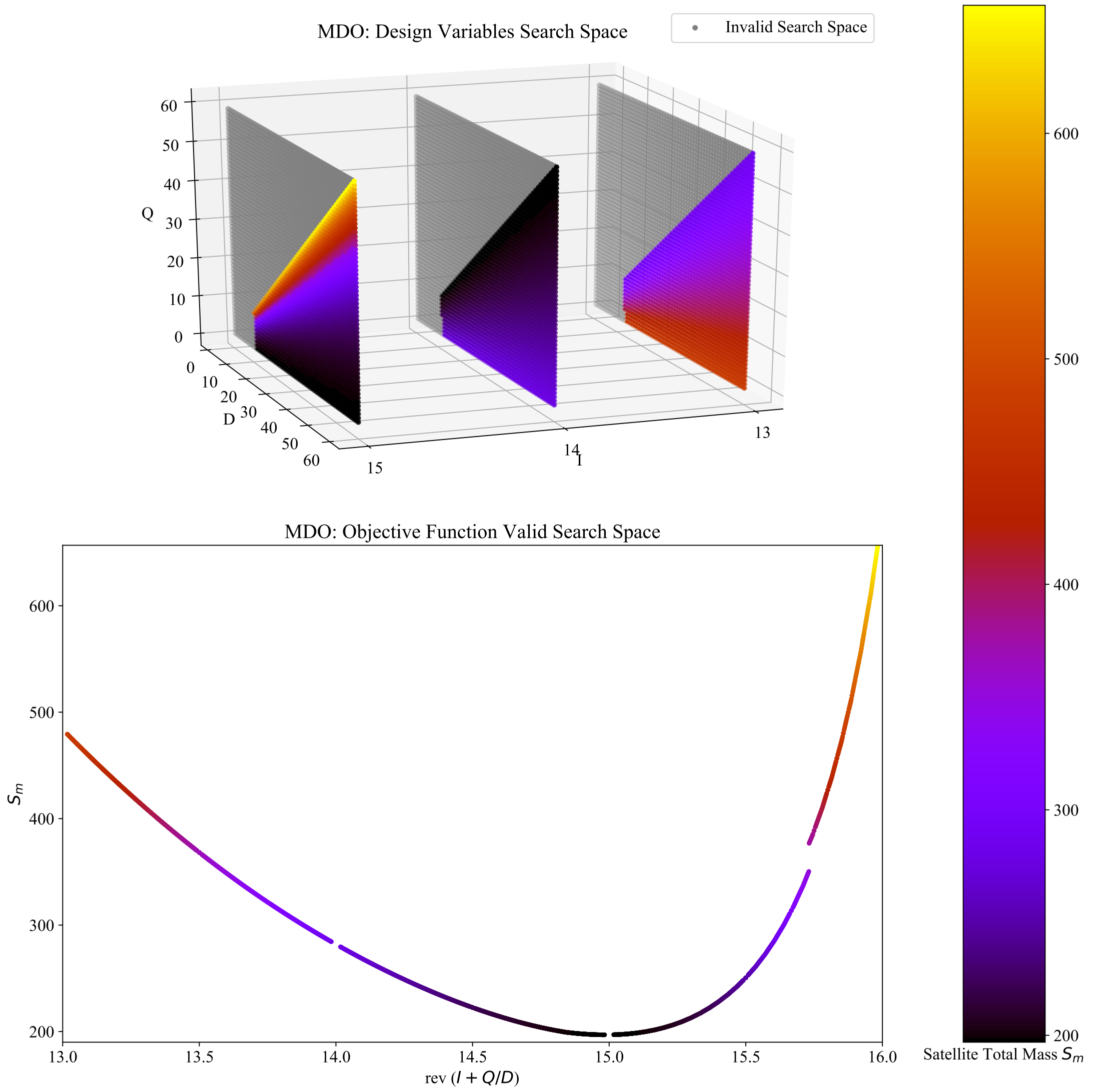
System Item	Value
Processor	Intel(R) Core(TM) i5-4300U CPU @ 1.9GHz 2.50GHz
RAM Memory	4,00 GB
System Type	64-bit Operating System, x64 based processor
Operating System	Windows 10
HD Type	SSD

SOURCE: Made by the author.

It was only possible to perform the extensive search because the global design variables (I, Q and D) have only 10,620 possible combinations and it is a simplified model. On real applications, the MDO would be composed of many other models that require orbit propagations to be evaluated. This leads to an algorithm with very high computational burden in which an extensive search is not feasible during the conceptual design phase timespan.

Since an extensive search was done, it was possible to map the entire search space for the problem, shown in Figure 5.5. The solution values S_{m} that violated the problem constrains are shown in gray and its values are not shown in the gradient color map.

Figure 5.5 - Spacecraft conceptual design search space.

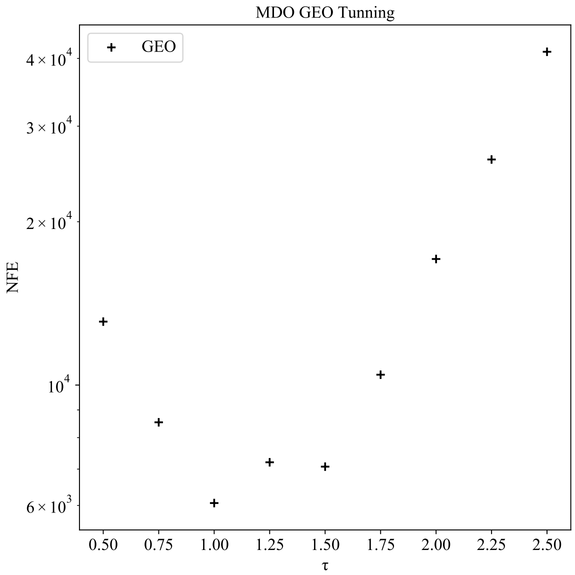


SOURCE: Made by the author.

With the data presented in Figure 5.5 it is clear that the problem has a search space extremely constrained, since almost half of it is composed of invalid regions, that are presented in all project space. Thus, making this problem a very constrained one.

Next, 100 independent runs were executed for each algorithm, using the global optimum as the stop criteria for the MDO simulations to evaluate the performance of the optimizers on the application. At each run the design variables were configured as shown in Table 5.1. For GEO, the tunning was performed a priori with τ ranging between $[0.5; 2.5]$ by steps of 0.25 as shown in Figure 5.6. It turns out that the best configuration was $\tau = 1$, which was selected to compare the results.

Figure 5.6 - Spacecraft conceptual design GEO Tunning.



SOURCE: Made by the author.

The results shown in Table 5.4 are the average NFE necessary to reach the optimal solution for the 100 independent algorithms runs, its standard deviation, coefficient of variation and the average runtime for one run. As can be seen, A-GEO$_2$ outperforms all algorithms and A-GEO$_1$ has the worse performance. Considering the evolutionary algorithms versus the extensive search A-GEO$_1$ reduces the computational cost by 3.99%, while GEO and A-GEO$_2$ reduced it by 42.93% and 46.18% respectively, which means a significant gain of performance without losing quality.

Further analyzing the algorithms performance, its noticeable that the evolutionary approaches are extremely better than the extensive search, especially, when it comes to the time needed to complete one full run of the simulations. Since all the evolutionary approaches take less Average Runtime(t) to be evaluated than the extensive search, due to its characteristic of at each run find a possible different path till the final solution, they can decrease the computational cost and reduce the time needed to complete the runs, while for the extensive search the computational cost in any run would be the same. By the way, another information provided by the average runtime is that A-GEO does not increase the complexity of the algorithm over GEO, since the difference between times can be computed by a linear equation that if also applied to the NFE generates a similar outcome.

Table 5.4 - Simulation results.

Algorithm	NFE	NFE σ	NFE c_v	Average Runtime(t)
Extensive search	1.062E+04	-	-	00:11:43
GEO ($\tau = 1$)	6.061E+03	5.575E+03	0.920E-01	00:01:39
A-GEO$_1$	1.020E+04	1.009E+04	0.990E-01	00:03:37
A-GEO$_2$	**5.716E+03**	**4.935E+03**	**0.860E-01**	**00:01:34**

SOURCE: Made by the author.

Finally, these performances improvement, especially the ones resulted from A-GEO$_2$, become even more significant when its considered calculations that involve complex simulations, as in solar power cycles of charge and discharge simulations for an orbit period. These types of simulations demand huge amounts of computational power that can be minimized using such types of algorithms as the one developed in this work.

6 CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

A new version of the GEO algorithm was introduced, named A-GEO. A-GEO is an adaptive evolutionary algorithm that eliminates the only free parameter presented in GEO, called τ, by employing an adaptive control mechanism to the algorithm, thus making the process of tuning GEO unnecessary. Without a parameter to be tuned, A-GEO becomes a parameterless version of GEO and shows the benefit of having the algorithm parameters being internally controlled, making easier and faster the work of the final user.

When A-GEO was compared with GEO it was made evident the gain in performance that can be obtained by the adaptive approach. In fact, A-GEO performed better than GEO in all numerical tests performed, except for F1. It is noteworthy that even when the best τ is known a priori, A-GEO can still have a better performance than GEO, although not for all test functions.

Two different implementations of A-GEO were developed in this work. They differ on how a reference population used to assess performance improvement is defined. While in A-GEO$_1$ it is used the best population found so far in the search, in A-GEO$_2$ the current population is used. A-GEO$_2$ performed better than A-GEO$_1$, and this difference seems to be the result of the way the algorithms balance the ratio between exploration and exploitation during the search. In A-GEO$_1$ there are more resets in the value of τ than in A-GEO$_2$, indicating that it is more prone to be stuck in local minimums than A-GEO$_2$.

From the point of view of the dispersion parameter C_v, which indicates how robust to different initializations the results of a given algorithm are, GEO, when set to the best τ, appears to be better than A-GEO. However, this indicates only that GEO tends to converge more rapidly to a given minimum, but it may not be the global optimum. Ideally the algorithm would converge to the global optimum area independently from which point (solution) the run was initialized. Using this concept, the convergence is desirable for an algorithm if it can delivery good solutions or if the problem has some design limitation that must assure that the solutions found must not be scattered through the search space. A-GEO resets make the algorithm explore other search space regions sacrificing its convergence, but delivering better results, which implies a compromise or trade off to balance its exploitation and exploration.

These resets are a important part of the free parameter behavior produced by A-GEO controller mechanism. It was observed that the algorithm made the free parameter more deterministic in the beginning of the search, while after this it oscillated inside a variable interval. These oscillations happened in cycles, which were shorter for A-GEO$_1$. This may indicate that A-GEO$_2$ has a better control on its exploration versus exploitation than A-GEO$_1$, since it performed less resets maintaining a more controlled value of τ through the search. More experiments shall be conducted following the present work, to study this behavior.

Finally, A-GEO$_2$ was compared with a recent developed evolutionary algorithm, named TBOL-FL, using a set of test functions from CEC2017. As expected, there is a clear performance gap between the two. A-GEO$_2$ performs poorly on unimodal and hybrid functions from CEC2017 benchmark and loses in overall performance for TBOL-FL on almost every one of the 29 functions tested, except one. However, this is an expected outcome considering that A-GEO$_2$ is a recent developed algorithm that did great improvements on GEO, an algorithm from almost 20 years ago. Thus, this is the first step towards a competitive state of art adaptive version of GEO. The fact that it beat a “top” evolutionary algorithm in one test function and on multimodal and composite functions exhibit competitive performance already shows that this work is in the right direction. Part of this gap in performance between theses algorithms can be caused by how A-GEO$_2$ design variables are encoded, since it uses binary encoding. Nonetheless, further improvements in A-GEO$_2$ performance may also be obtained by using other strategies for its control mechanism.

After the comparisons against GEO and TLBO-FL, A-GEO was used as an optimizer in a MDO to solve an space application optimization problem of conceptual satellite design with a constrained search space, were A-GEO$_2$ performed better than GEO and reduced the problem computational cost in almost half. All this implies that A-GEO has a great potential to be applied in real engineering and scientific problems, specially of the user-friendly point of view, promising to be a valuable tool to be incorporated to the engineer’s or scientist’s toolbox. However, first the algorithm has to be better worked on and improved to fulfill its gap to the current generation of evolutionary algorithms.

Thus, the author proposes that future work should be directed in:

- Implement A-GEO with real encoding;
- Improve the control parameter technique presented on A-GEO;

- Apply the control parameter technique presented on A-GEO to other algorithms;
- Investigate the A-GEO behavior through the search space to confirm the relation presented between the algorithm performance and the balance of exploration and exploitation, and;
- Develop other adaptive or self adaptive versions of GEO.

REFERENCES

AHRARI, A.; SHARIAT-PANAHI, M. An improved evolution strategy with adaptive population size. **Optimization**, v. 64, n. 12, p. 2567–2586, 2015. Available from: <https://doi.org/10.1080/02331934.2013.836651>. 5

ALBUQUERQUE, B. F. C. d.; SOUSA, F. L. D.; MONTES, A. S. Multi-objective approach for the automatic design of optical systems. **Optics Express**, v. 24, n. 6, p. 6619–6643, Mar 2016. 11

ALETI, A.; MOSER, I. A systematic literature review of adaptive parameter control methods for evolutionary algorithms. **ACM Computing Surveys**, v. 49, n. 3, p. 56:1–56:35, oct. 2016. ISSN 0360-0300. xv, 3, 5, 6, 7, 8, 9, 15, 18, 19, 20

ALETI, A.; MOSER, I.; MOSTAGHIM, S. Adaptive range parameter control. In: CONGRESS ON EVOLUTIONARY COMPUTATION, 2012. **Proceedings...** [S.l.]: IEEE. p. 1–8. ISSN 1089-778X. 5

AWAD, N. H.; ALI, M.; LIANG, J. J.; QU, B. Y.; SUGANTHAN, P. N. **Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization**. [S.l.], 2017. 10, 43, 45

BAK, P.; CHEN, K. Self-organized criticality. **Scientific American**, v. 264, n. 1, p. 46–53, 1991. 13

BOETTCHER, S.; PERCUS, A. G. Optimization with extremal dynamics. **Physical Review Letters**, v. 86, p. 5211–5214, Jun 2001. 11

CERVANTES, J.; STEPHENS, C. R. Limitations of existing mutation rate heuristics and how a rank ga overcomes them. **IEEE Transactions on Evolutionary Computation**, v. 13, n. 2, p. 369–397, Apr. 2009. ISSN 1089-778X. 3, 5

CHAGAS, R. A. J.; ALBUQUERQUE, B. F. C. de; LOPES, R. A. M.; SOUSA, F. L. D. Towards the automation of concurrent space systems conceptual design through multidisciplinary design. In: ABCM INTERNATIONAL CONGRESS OF MECHANICAL ENGINEERING, 23., 2015. **Proceedings...** [S.l.]. 1, 55, 56, 59, 60, 61, 63, 64, 65, 68

CHAGAS, R. A. J.; GALSKI, R. L.; SOUSA, F. L. D. An orbit selection tool for satellite constellations using the Multiobjective Generalized Extremal Optimization (MGEO) algorithm. In: INTERNATIONAL SYSTEMS AND CONCURRENT ENGINEERING FOR SPACE APPLICATIONS CONFERENCE, 6., 2014. **Proceedings...** [S.l.]. 1, 55

CHAGAS, R. A. J.; LOPES, R. **Amazonia-1 orbit analysis report**. [S.l.], 2015. 57

CHEN, M. R.; LU, Y. Z.; YANG, G. Population-based extremal optimization with adaptive lévy mutation for constrained optimization. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL INTELLIGENCE AND SECURITY, 2006. **Proceedings...** [S.l.], 2006. v. 1, p. 258–261. 13

COELHO, L. d. S.; MARIANI, V. C.; GREBOGI, R. B.; SEGUNDO, E. H. de V.; LUZ, M. V. F. da; LEITE, J. V.; FREIRE, R. Z. Diversity-guided generalized extremal optimization for transformer design problem. In: SYMPOSIUM SERIES ON COMPUTATIONAL INTELLIGENCE, 2017. **Proceedings...** [S.l.]: IEEE, 2017. p. 1–6. 11

CUCO, A. P. C.; SILVA NETO, A. J.; CAMPOS VELHO, H. F.; SOUSA, F. L. D. Solution of an inverse adsorption problem with an epidemic genetic algorithm and the generalized extremal optimization algorithm. **Inverse Problems in Science and Engineering**, v. 17, n. 3, p. 289–302, 2009. 11

DARWIN, C. **On the origin of species by means of natural selection**. London: Murray, 1859. 1

DERRAC, J.; GARCíA, S.; MOLINA, D.; HERRERA, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. **Swarm and Evolutionary Computation**, v. 1, n. 1, p. 3 – 18, 2011. ISSN 2210-6502. Available from: <http://www.sciencedirect.com/science/article/pii/S2210650211000034>. 48

DORIGO, M.; CARO, G. D.; GAMBARDELLA, L. M. Ant algorithms for discrete optimization. **Artificial Life**, v. 5, n. 2, p. 137–172, apr. 1999. ISSN 1064-5462. Available from: <http://dx.doi.org/10.1162/106454699568728>. 1

EIBEN, A.; SMIT, S. Parameter tuning for configuring and analyzing evolutionary algorithms. **Swarm and Evolutionary Computation**, v. 1, n. 1, p. 19 – 31, 2011. ISSN 2210-6502. 3, 4

EIBEN, A. E.; HINTERDING, R.; MICHALEWICZ, Z. Parameter control in evolutionary algorithms. **IEEE Transactions on Evolutionary Computation**, v. 3, n. 2, p. 124–141, July 1999. ISSN 1089-778X. 3, 5, 9, 25

EIBEN, A. E.; SMITH, J. From evolutionary computation to the evolution of things. **Nature**, v. 521, p. 476–482, 2015. 1

EIBEN, A. E.; SMITH, J. E. **Introduction to evolutionary computing**. [S.l.]: Berlin: Springer, 2003. ISSN 1619-7127. 1, 17

FLIEGE, J.; KAPARIS, K.; KHOSRAVI, B. Operations research in the space industry. **European Journal of Operational Research**, v. 217, n. 2, p. 233–240, 2012. ISSN 0377-2217. 1

FREITAS, V. L.; SOUSA, F. L. D.; MACAU, E. E. Reactive model for autonomous vehicles formation following a mobile reference. **Applied Mathematical Modelling**, v. 61, p. 167 – 180, 2018. ISSN 0307-904X. 11

GALSKI, R.; JÚNIOR, H. P.; SOUSA, F. L. D.; HINCKEL, J.; LACAVA, P.; RAMOS, F. M. GEO + ES Hybrid optimization algorithm applied to the parametric thermal model estimation of a 200N hydrazine thruster. In: INTERNATIONAL DESIGN ENGINEERING TECHNICAL CONFERENCES AND COMPUTERS AND INFORMATION IN ENGINEERING CONFERENCE, 2011. **Proceedings...** [S.l.]: ASME. v. 2, p. 407–414. ISBN 9781467312127. 11, 15

GALSKI, R. L. **Desenvolvimento de versões aprimoradas híbridas, paralela e multiobjetivo do método da otimização extrema generalizada e sua aplicação no projeto de sistemas espaciais**. 279 p. INPE-14795-TDI/1238. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2007. 11, 13, 14

GALSKI, R. L.; SOUSA, F. L. D.; RAMOS, F. M.; SILVA NETO, A. J. Application of a GEO + SA hybrid optimization algorithm to the solution of an inverse radiative transfer problem. **Inverse Problems in Science and Engineering**, v. 17, n. 3, p. 321–334, 2009. 11, 14, 15

GINLEY, B. M.; MAHER, J.; O'RIORDAN, C.; MORGAN, F. Maintaining healthy population diversity using adaptive crossover, mutation, and selection. **IEEE Transactions on Evolutionary Computation**, v. 15, n. 5, p. 692–714, Oct 2011. ISSN 1089-778X. 5

GOLDBERG, D. E. **Genetic algorithms in search, optimization and machine learning**. Boston, MA, USA: [s.n.], 1989. ISBN 0201157675. 1, 12

GOLDMAN, B. W.; TAURITZ, D. R. Meta-evolved empirical evidence of the effectiveness of dynamic parameters. In: ANNUAL CONFERENCE COMPANION ON GENETIC AND EVOLUTIONARY COMPUTATION, 13., 2011, New York, USA. **Proceedings...** [S.l.]: ACM, 2011. (GECCO '11), p. 155–156. ISBN 978-1-4503-0690-4. 3

GOSSELIN, L.; TYE-GINGRAS, M.; MATHIEU-POTVIN, F. Review of utilization of genetic algorithms in heat transfer problems. **International Journal of Heat and Mass Transfer**, Elsevier Ltd, v. 52, n. 9-10, p. 2169–2188, 2009. ISSN 0017-9310. 1

HOLLAND, J. H. **Adaptation in natural and artificial system**. [S.l.]: MIT Press, 1975. 1

JITKONGCHUEN, D.; THAMMANO, A. A self-adaptive differential evolution algorithm for continuous optimization problems. **Artificial Life and Robotics**, v. 19, n. 2, p. 201–208, 2014. 5

KARAFOTIAS, G.; HOOGENDOORN, M.; EIBEN, A. E. Parameter control in evolutionary algorithms: trends and challenges. **IEEE Transactions on Evolutionary Computation**, v. 19, n. 2, p. 167–187, Apr. 2015. ISSN 1089-778X. 2, 3, 4, 5, 6, 15, 18

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: INTERNATIONAL CONFERENCE ON NEURAL NETWORKS, 4., 1995. [S.l.], 1995. v. 4, p. 1942–1948. 1

KIRKPATRICK, S.; Gellat JUNIOR, C. D.; VECCHI, M. P. Optimization by simulated annealing. **Science**, v. 220, n. 4598, p. 671–680, 1983. 15

Kommadath, R.; Kotecha, P. Teaching learning based optimization with focused learning and its performance on cec2017 functions. In: CONGRESS ON EVOLUTIONARY COMPUTATION, 10., 2003, IEEE. **Proceedings...** [S.l.], 2017. p. 2397–2403. 10, 44, 45, 46, 47, 50, 52

KUMAR, S. R.; KURIAN, C. P.; GOMES-BORGES, M. E. Multiobjective generalized extremal optimization algorithm for simulation of daylight illuminants. **Journal of Photonics for Energy**, v. 7, p. 7–16, 2017. 11

LAMBE, A. B.; MARTINS, J. R. R. A. Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes. **Structural and Multidisciplinary Optimization**, v. 46, p. 273–284, 2012. 66

LIAN, Y.; OYAMA, A.; LIOU, M.-s. Progress in design optimization using evolutionary algorithms for aerodynamic problems. **Progress in Aerospace Sciences**, v. 46, n. 5-6, p. 199–223, 2010. ISSN 0376-0421. 1

LOBO, F. G.; LIMA, C. F. Adaptive population sizing schemes in genetic algorithms. In: LOBO, F. G.; LIMA, C. F.; MICHALEWICZ, Z. (Ed.). **Parameter setting in evolutionary algorithms**. Berlin, Heidelberg: Springer, 2007. p. 185–204. ISBN 978-3-540-69432-8. 5

LOPES, I. M. L. **Uma abordagem multi-objetivo para a otimização de trajetórias de uma vela solar**. 110 p. Tese (Doutorado em Engenharia e Tecnologia Espaciais / Mecânica Espacial e Controle) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2013. 13

LOPES, I. M. L.; SOUSA, F. L. D.; SOUZA, L. C. G. A comparative study between control strategies for a solar sailcraft in an Earth-Mars transfer. **Mechanical Systems and Signal Processing**, v. 79, p. 289–296, 2016. 11, 13

LOVIE, P. **Coefficient of variation**. [S.I.]: American Cancer Society, 2005. ISBN 9780470013199. Available from: <https://onlinelibrary.wiley.com/doi/abs/10.1002/0470013192.bsa107>. 28

MAINENTI-LOPES, I.; SOUZA, L.; SOUSA, F. L. D. Design of a nonlinear controller for a rigid-flexible satellite using multi-objective generalized extremal optimization with real codification. **Shock and Vibration**, v. 19, n. 5, p. 947 – 956, 2012. 11

MARTINS, J. R. R. A.; LAMBE, A. B. Multidisciplinary design optimization : a survey of architectures. **AIAA Journal**, v. 51, n. 9, p. 2049–2075, 2013. 1, 66

MURAOKA, I.; GALSKI, R. L.; SOUSA, F. L. D.; RAMOS, F. M. Stochastic spacecraft thermal design optimization with low computational cost. **Journal of Spacecraft and Rockets**, v. 43, n. 6, p. 1248–1257, 2006. 11

NADI, F.; KHADER, A. T. A parameter-less genetic algorithm with customized crossover and mutation operators. In: ANNUAL CONFERENCE ON GENETIC

AND EVOLUTIONARY COMPUTATION, 13., 2011, Dublin, Ireland. **Proceedings...** [S.l.]: ACM, 2011. p. 901–908. ISBN 978-1-4503-0557-0. 5

RAO, R.; SAVSANI, V.; VAKHARIA, D. Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. **Computer-Aided Design**, v. 43, n. 3, p. 303 – 315, 2011. ISSN 0010-4485. Available from: <http://www.sciencedirect.com/science/article/pii/S0010448510002484>. 44, 45

SCHMIDT, M.; LIPSON, H. Distilling free-form natural laws from experimental data. **Science**, v. 324, n. 5923, p. 81–85, 2009. ISSN 0036-8075. 1

SHESKIN, D. **Handbook of parametric and nonparametric statistical procedures**. 4. ed. [S.l.]: Chapman&Hall/CRC, 2006. 47

SHIRAZI, A. Trajectory optimization of spacecraft high-thrust orbit transfer using a modified evolutionary algorithm. **Engineering Optimization**, v. 48, n. 10, p. 1639–1657, 2015. 1

SILVA NETO, A. J. D.; BECCENERI, J. C.; CAMPOS VELHO, H. F. d. Otimização Extrema Generalizada (Generalized Extremal Optimization). In: SILVA NETO, A. J. D.; BECCENERI, J. C.; CAMPOS VELHO, H. F. d. (Ed.). **Inteligência computacional aplicada a problemas inversos em transferência radiativa**. [S.l.: s.n.], 2016. chapter 11, p. 155–170. 12, 13

SMITH, J.; FOGARTY, T. C. Self adaptation of mutation rates in a steady state genetic algorithm. In: INTERNATIONAL CONFERENCE ON EVOLUTIONARY COMPUTATION, 2003. **Proceedings...** [S.l.]: IEEE, 1996. p. 318–323. 3

SOUSA, F. L. D. **Otimização extrema generalizada: um novo algorítmo estocástico para o projeto ótimo**. 142 p. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, SP, Brazil, 2003. 1, 12, 13

SOUSA, F. L. D.; RAMOS, F. M. New stochastic algorithm for design optimization. **AIAA Journal**, v. 41, n. 9, p. 1808–1818, 2003. 1, 11, 12, 13

SOUSA, F. L. D.; RAMOS, F. M.; GALSKI, R. L.; MURAOKA. Generalized Extremal Optimization: A new Meta-Heuristic Inspired by a Model of Natural Evolution. In: CASTRO, L. N. d.; ZUBEN, F. J. V. (Ed.). **Recent developments in biologically inspired computing**. [S.l.]: Idea Group, 2005. chapter 3, p. 41–60. 12, 13

SWITALSKI, P.; SEREDYNSKI, F. Multiprocessor task scheduling based on geo metaheuristic. **Intelligent Information Systems**, p. 111–120, 2008. 11

VAFAEE, F.; NELSON, P. C. An explorative and exploitative mutation scheme. In: CONGRESS ON EVOLUTIONARY COMPUTATION, 2010. **Proceedings...** [S.l.]: IEEE, 2010. p. 1–8. ISSN 1089-778X. 5

VALLADO, D. A. **Fundamentals of astrodynamics and applications**. 3. ed. California, United States of America: Microcosm Press/Springer, 2007. 56

VLASSOV, V. V.; SOUSA, F. L. de; TAKAHASHI, W. K. Comprehensive optimization of a heat pipe radiator assembly filled with ammonia or acetone. **International Journal of Heat and Mass Transfer**, v. 49, n. 23, p. 4584 – 4595, 2006. ISSN 0017-9310. 11

WERTZ, J. R.; EVERETT, D.; PUSCHELL, J. **Space mission engineering: the new SMAD**. California, United States of America: Microcosm Press, 2011. 1, 55, 56, 58, 59, 60, 63, 64, 65, 68

WONG, Y.-Y.; LEE, K.-H.; LEUNG, K.-S.; HO, C.-W. A novel approach in parameter adaptation and diversity maintenance for genetic algorithms. **Soft Computing**, v. 7, n. 8, p. 506–515, Aug 2003. ISSN 1432-7643. Available from: <https://doi.org/10.1007/s00500-002-0235-1>. 6

XIE, D.; LUO, Z.; YU, F. The computing of the optimal power consumption for semi-track air-cushion vehicle using hybrid generalized extremal optimization. **Applied Mathematical Modelling**, v. 33, n. 6, p. 2831 – 2844, 2009. ISSN 0307-904X. 11, 13

ZENG, G.-q.; LU, K.-d.; CHEN, J.; ZHANG, Z.-j.; DAI, Y.-x.; PENG, W.-w.; ZHENG, C.-w. An improved real-coded population-based extremal optimization method for continuous unconstrained optimization problems. **Mathematical Problems in Engineering**, v. 2014, p. 9, 2014. 13

PUBLICAÇÕES TÉCNICO-CIENTÍFICAS EDITADAS PELO INPE

Teses e Dissertações (TDI)

Teses e Dissertações apresentadas nos Cursos de Pós-Graduação do INPE.

Notas Técnico-Científicas (NTC)

Incluem resultados preliminares de pesquisa, descrição de equipamentos, descrição e ou documentação de programas de computador, descrição de sistemas e experimentos, apresentação de testes, dados, atlas, e documentação de projetos de engenharia.

Propostas e Relatórios de Projetos (PRP)

São propostas de projetos técnico-científicos e relatórios de acompanhamento de projetos, atividades e convênios.

Publicações Seriadas

São os seriados técnico-científicos: boletins, periódicos, anuários e anais de eventos (simpósios e congressos). Constam destas publicações o Internacional Standard Serial Number (ISSN), que é um código único e definitivo para identificação de títulos de seriados.

Pré-publicações (PRE)

Todos os artigos publicados em periódicos, anais e como capítulos de livros.

Manuais Técnicos (MAN)

São publicações de caráter técnico que incluem normas, procedimentos, instruções e orientações.

Relatórios de Pesquisa (RPQ)

Reportam resultados ou progressos de pesquisas tanto de natureza técnica quanto científica, cujo nível seja compatível com o de uma publicação em periódico nacional ou internacional.

Publicações Didáticas (PUD)

Incluem apostilas, notas de aula e manuais didáticos.

Programas de Computador (PDC)

São a seqüência de instruções ou códigos, expressos em uma linguagem de programação compilada ou interpretada, a ser executada por um computador para alcançar um determinado objetivo. Aceitam-se tanto programas fonte quanto os executáveis.