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

sid.inpe.br/mtc-m21d/2022/07.08.16.37-TDI

ESTUDO PARA AUMENTO DE DESEMPENHO DE VERSÃO ADAPTATIVA DO ALGORITMO GEO E SUA APLICAÇÃO NO PROJETO CONCEITUAL DE SISTEMAS ESPACIAIS

Leonardo Becker da Luz

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais, orientada pelos Drs. Fabiano Luis de Sousa, e Ronan Arraes Jardim Chagas, aprovada em 30 de junho de 2022.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34T/478GL8H>

INPE
São José dos Campos
2022

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Coordenação de Ensino, Pesquisa e Extensão (COEPE)
Divisão de Biblioteca (DIBIB)
CEP 12.227-010
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/7348
E-mail: pubtc@inpe.br

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

sid.inpe.br/mtc-m21d/2022/07.08.16.37-TDI

ESTUDO PARA AUMENTO DE DESEMPENHO DE VERSÃO ADAPTATIVA DO ALGORITMO GEO E SUA APLICAÇÃO NO PROJETO CONCEITUAL DE SISTEMAS ESPACIAIS

Leonardo Becker da Luz

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais, orientada pelos Drs. Fabiano Luis de Sousa, e Ronan Arraes Jardim Chagas, aprovada em 30 de junho de 2022.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34T/478GL8H>

INPE
São José dos Campos
2022

Dados Internacionais de Catalogação na Publicação (CIP)

Luz, Leonardo Becker da.

L979e Estudo para aumento de desempenho de versão adaptativa do algoritmo geo e sua aplicação no projeto conceitual de sistemas espaciais / Leonardo Becker da Luz. – São José dos Campos : INPE, 2022.

xxiv + 115 p. ; (sid.inpe.br/mtc-m21d/2022/07.08.16.37-TDI)

Dissertação (Mestrado em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2022.

Orientadores : Drs. Fabiano Luis de Sousa, e Ronan Arraes Jardim Chagas.

1. Algoritmos evolutivos. 2. Controle de parâmetros. 3. Sistemas espaciais. I.Título.

CDU 629.78:004.421.2

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÕES

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS
Serviço de Pós-Graduação - SEPGR

DEFESA FINAL DE DISSERTAÇÃO LEONARDO BECKER DA LUZ
BANCA N°193/2022, REG. 32070/2020.

No dia 30 de junho de 2022, as 09h, por teleconferência, o(a) aluno(a) mencionado(a) acima defendeu seu trabalho final (apresentação oral seguida de arguição) perante uma Banca Examinadora, cujos membros estão listados abaixo. O(A) aluno(a) foi APROVADO(A) pela Banca Examinadora, por unanimidade, em cumprimento ao requisito exigido para obtenção do Título de Mestre em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais. O trabalho precisa da incorporação das correções sugeridas pela Banca e revisão final pelo(s) orientador(es).

Título: "ESTUDO PARA AUMENTO DE DESEMPENHO DE VERSÃO ADAPTATIVA DO ALGORITMO GEO E SUA APLICAÇÃO NO PROJETO CONCEITUAL DE SISTEMAS ESPACIAIS".

Membros da Banca:

Dr. Walter Abrahao Dos Santos - Presidente - INPE
Dr. Fabiano Luis de Sousa - Orientador - INPE
Dr. Ronan Arraes Jardim Chagas - Orientador - INPE
Dr. Valdivino Alexandre de Santiago Júnior - Membro Interno - INPE
Dr. Antônio Augusto Chaves - Membro Externo - UNIFESP

Documento assinado eletronicamente por **Valdivino Alexandre de Santiago Júnior**, **Tecnologista em Ciência e Tecnologia**, em 06/07/2022, às 15:19 (horário oficial de Brasília), com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Ronan Arraes Jardim Chagas**, **Tecnologista**, em 06/07/2022, às 17:37 (horário oficial de Brasília), com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Fabiano Luís de Sousa**, **Tecnologista**, em 07/07/2022, às 10:13 (horário oficial de Brasília), com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Walter Abrahão dos Santos**, **Tecnologista em Ciência e Tecnologia**, em 08/07/2022, às 09:35 (horário oficial de Brasília), com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Antonio Augusto Chaves (E)**, **Usuário Externo**, em 13/07/2022, às 13:54 (horário oficial de Brasília), com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site http://sei.mctic.gov.br/verifica.html, informando o código verificador **10102284** e o código CRC **F6CCFBF4**.

Referência: Processo nº 01340.005094/2022-39 SEI nº 10102284

“The greatest enemy of knowledge is not ignorance, it is the illusion of knowledge.”

Stephen Hawking

Dedico este trabalho à minha família, namorada, pais e amigos, que sempre estiveram dando apoio e não mediram esforços para que eu chegasse até esta etapa importante.

AGRADECIMENTOS

Ao meu orientador, Fabiano, pela oportunidade de trabalharmos juntos, tornando a minha experiência no mestrado um grande aprendizado na área. Por todo o apoio e troca de ideias nesse período que convivemos e, também, por incentivar e me ajudar a evoluir como indivíduo. Grande parte do meu crescimento devo a ele.

Ao meu coorientador, Ronan, por todas as conversas e aprendizados. Por ter um mundo de conhecimentos a difundir e por me ajudar a crescer não só academicamente, mas pessoalmente. Agradeço também pela amizade e pelo suporte.

Aos meus pais, Luís e Denise, que dedicaram a vida à educação dos filhos. Obrigado por sempre me incentivarem a conquistar meus objetivos e por me darem suporte em todos momentos. Quero agradecê-los por todo amor, carinho, compreensão e por serem tão especiais. É muito bom saber que posso contar com eles em todos momentos.

Aos meus irmãos e cunhadas, Leandro, Letícia, Eduardo e Isa, que sempre me apoiam e torcem pela realização dos meus sonhos. Obrigado por sempre me apoiarem, motivarem a seguir em frente e por serem pessoas que eu posso contar em todos os momentos.

Agradeço a todos os amigos e colegas que fizeram parte desta etapa, e que, de alguma forma, contribuem para tornar-me melhor. Em especial, aos meus amigos, Samuel, Lucas, Mateus, Luíza, Augusto, Daniel, Renata, Deborah e vários outros, que sempre estiveram me dando força para enfrentar os momentos difíceis.

Quero muito agradecer a minha segunda família, Rodigheri, por tornarem minha caminhada mais feliz e completa. Agradeço por todo carinho, aprendizado, apoio e por todos os momentos bons que eles têm me proporcionado.

A todos os professores, colegas e amigos da ETE, do BDC e do SERE por contribuírem para meu crescimento pessoal e profissional.

Agradeço ao INPE e a CAPES pela oportunidade de realizar um curso de Pós-Graduação, com incentivo a congressos, ampliando o conhecimento e as experiências de vida.

Por fim, e não menos importante, quero agradecer a duas pessoas que são muito próximas e importantes na minha vida.

A minha cunhada, Mari, por ser tão atenciosa e querida, sempre torcendo para o meu crescimento pessoal e profissional e me ajudando. Por ser um exemplo de pessoa pra mim e por me mostrar que os sonhos são alcançados com determinação. Obrigado por todos os momentos bons.

A minha companheira de vida, meu maior exemplo, e a principal razão de eu estar feliz na área acadêmica, Grazi. Obrigado por todos os momentos que passamos juntos e por me ensinar que as coisas mais simples da vida são as que proporcionam os melhores momentos. Agradeço por toda ajuda e por toda a parceria durante todos os dias em que vivemos juntos, onde eu tenho a oportunidade de evoluir e compartilhar dos sentimentos mais nobres. Obrigado pelo companheirismo para enfrentarmos os momentos bons e difíceis.

RESUMO

O desenvolvimento de algoritmos evolutivos adaptativos para problemas de otimização é desafiador, uma vez que estabelecer uma abordagem de controle de parâmetros eficiente pode tornar-se uma tarefa difícil, principalmente se houver uma grande quantidade de parâmetros livres ou se estes forem interdependentes. Por outro lado, pode ser desenvolvida uma ferramenta muito eficiente na busca pelo ótimo no espaço de projeto e sem a necessidade de ajuste de parâmetros. Recentemente, foi proposta uma versão adaptativa do algoritmo Otimização Extrema Generalizada, denominada A-GEO, que em seus primeiros testes apresentou um melhor desempenho e facilidade de utilização do que o seu antecessor não adaptativo. Com base nisso, várias novas implementações do A-GEO foram propostas neste trabalho utilizando codificação binária e real para as variáveis de projeto, sendo uma delas uma versão semi auto-adaptativa. Seus desempenhos foram avaliados por meio de funções teste com diferentes características morfológicas. As implementações de melhor resultado foram avaliadas utilizando um conjunto maior de funções da competição CEC 2017 Real-Parameter Single Objective Optimization e aplicadas em um problema de otimização na fase conceitual de uma missão espacial. Os resultados indicaram que, para problemas com espaço de projeto contínuo e somente restrições laterais, a utilização da codificação real pode levar a um ganho significativo de desempenho em comparação a codificação binária. Além disso, foi constatado que ganhos de desempenho significativos dependem de uma correspondência adequada do algoritmo com o problema de otimização.

Palavras-chave: Algoritmos Evolutivos. Controle de Parâmetros. Sistemas Espaciais.

STUDY TO INCREASE THE PERFORMANCE OF THE ADAPTIVE VERSION OF THE GEO ALGORITHM AND ITS APPLICATION IN THE CONCEPTUAL PROJECT OF SPACE SYSTEMS

ABSTRACT

The development of adaptive evolutionary algorithms for optimization problems is challenging, since establishing an efficient parameter control approach can become a difficult task, especially if there are many free parameters or if they are entangled. On the other hand, a tool can be developed that is very efficient in finding the optimum in the design space and without the need for parameter tuning. Recently, an adaptive version of the Generalized Extremal Optimization algorithm, called A-GEO, was proposed, which in its first tests showed better performance and usability than its non-adaptive predecessor. Several new implementations of A-GEO were proposed in this work using binary and real coding for the design variables, one of them being a semi-self-adaptive version. Their performances were evaluated using test functions with different morphological characteristics. The best performing implementations were evaluated using a larger set of functions from the CEC 2017 Real-Parameter Single Objective Optimization competition and applied to an optimization problem in the conceptual phase of a space mission. The results indicated that, for problems with continuous design space and only lateral constraints, using real encoding can lead to a significant performance gain compared to binary encoding. In addition, it was found that significant performance gains depend on proper matching of the algorithm to the optimization problem.

Keywords: Evolutionary Algorithms. Parameter Control. Space Systems.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE SIGLAS E ABREVIATURAS

AE	Algoritmos Evolutivos
AG	Algoritmos Genético
A-GEO	Adaptive Generalized Extremal Optimization / Otimização Extrema Generalizada Adaptativo
CEC	Congress on Evolutionary Computation
CoI	Chance of Improvement / Chance de Melhoramento
CPRIME	Centro de Projetos Integrado de Missões Espaciais
DISEP	Divisão de Sistemas Espaciais
DS	Método Divisão por S
EO	Extremal Optimization / Otimização Extrema
FOV	Field of View / Campo de Visão
GEO	Generalized Extremal Optimization / Otimização Extrema Generalizada
INPE	Instituto Nacional de Pesquisas Espaciais
LEO	Low Earth Orbit / Órbitas Terrestres Baixas
MDF	Multidisciplinary Feasible / Métodos Multidisciplinares
MDO	Multidisciplinary Design Optimization / Otimização de Projeto Multidisciplinar
MEO	Medium Earth Orbit / Órbitas Terrestres Médias
MOGA	Multi-objecitive Genetic Algorithm (MOGA)
NFE	Number of Function Evaluations / Número de Avaliações da Função Objetivo
ReCon	Reconfigurable satellite Constellation
SA	Simulated Annealing / Recozimento Simulado
SCGA	Structured-Chromosome Genetic Algorithm / Algoritmo Genético de Cromossomos Estruturados
SID	Serviço de Informação e Documentação
SSGRO	Sun Synchronuos Ground Repeating Orbit / Órbita de repetição terrestre heliossíncrona
VO	Método Variação Original

LISTA DE SÍMBOLOS

τ	Parâmetro livre tau
σ	Parâmetro desvio-padrão dos algoritmos com codificação real
ρ	Parâmetro porcentagem dos algoritmos com codificação real

SUMÁRIO

Pág.

1 INTRODUÇÃO

Na fase conceitual de desenvolvimento de uma nova espaçonave, são avaliadas diferentes soluções possíveis para suas arquiteturas elétricas e mecânicas, buscando encontrar aquela que mais se adapta a missão da espaçonave (LAU et al., 2014), considerando o desempenho e o custo de fabricação (BOUDJEMAI et al., 2007). No entanto, este processo requer conhecimento e experiência e é longo e caro em termos de mão de obra (KALITA; THANGAVELAUTHAM, 2020). Além disso, a falta de uma abordagem sistemática para avaliar totalmente todo o espaço de projeto poderia levar a uma solução abaixo do ideal ou, pior, a uma solução intratável, acarretando em uma possível perda de soluções inovadoras (KALITA; THANGAVELAUTHAM, 2020). Assim, qualquer automação numérica desse processo que possa criar e avaliar soluções candidatas, reduzir os custos e aumentar a qualidade do conceito de missão é altamente desejável (HINCKLEY et al., 2017).

Uma ferramenta com grande potencial para auxiliar o time de projeto é a otimização numérica, que faz a busca por soluções ótimas de acordo com objetivos e restrições formulados matematicamente (MUELLER; OCHSENDORF, 2015). A otimização de projeto por meio de computação evolutiva tem mostrado vantagens competitivas em termos de robustez, desempenho e criatividade e tem sido estudada há décadas (KALITA; THANGAVELAUTHAM, 2020).

Devido ao tamanho e complexidade dos sistemas aeroespaciais, o projeto de um sistema inteiro é dividido em várias disciplinas e uma das técnicas utilizadas para resolver tais problemas é a de Otimização de Projeto Multidisciplinar (MDO), que une métodos de otimização à modelos matemáticos de áreas distintas de um projeto para encontrar soluções de compromisso. Todavia, ao manusear uma série de disciplinas ao mesmo tempo, há um aumento significativo da complexidade do problema a ser resolvido e na quantidade de trade-offs.

Os avanços no uso de algoritmos evolutivos e meta-heurísticas inspiradas na natureza em aplicações de engenharia trazem uma oportunidade e um desafio

para os pesquisadores avançarem na otimização de produtos, sistemas e serviços (GREINER et al., 2018). Um desses algoritmos baseados na seleção natural é o algoritmo de Otimização Extrema Generalizada (GEO - Generalized Extremal Optimization) (DE SOUSA, 2002; DE SOUSA et al., 2003). Esse algoritmo é uma generalização do método de Otimização Extrema (EO - Extremal Optimization) proposto por (BOETTCHER; PERCUS, 2001). O GEO pode ser aplicado a problemas com espaços de projeto não convexos ou desconexos e com variáveis contínuas, discretas ou inteiras, enquanto possui apenas um parâmetro livre a ser ajustado (DE SOUSA et al., 2003).

Pesquisas foram realizadas com o GEO a fim de melhorar sua eficiência e foram desenvolvidas diversas versões do mesmo, como híbrida (GALSKI et al., 2011), paralela (GALSKI, 2006) e multiobjetivo com codificação real (MAINENTI-LOPES et al., 2012). O GEO e suas versões vem sendo aplicadas com sucesso em muitos problemas de otimização em engenharia, como projeto de trajetórias e sistemas de espaçonaves (MURAOKA et al., 2006; MAINENTI-LOPES et al., 2016), transferência radiativa (DE SOUSA et al. 2007), projeto de sistemas ópticos (DE ALBUQUERQUE et al., 2016) e veículos autônomos (FREITAS et al., 2018).

Uma vantagem do GEO sobre outros algoritmos evolutivos é que ele tem apenas um parâmetro livre para ajuste. Embora seja uma prática comum em computação evolutiva, o ajuste de parâmetros pode acarretar em desvantagens, como a necessidade de realizar uma enorme quantidade de experimentos para se obter um conjunto de valores que leve ao melhor desempenho do algoritmo para uma dada aplicação (EIBEN et al., 1999).

Uma versão adaptativa do GEO foi desenvolvida com um mecanismo de controle de parâmetros por Barroca (2019), sendo ela denominada A-GEO. Essa versão apresentou um desempenho superior ao GEO para um conjunto de testes numéricos. Apesar do A-GEO ser um algoritmo de otimização sem parâmetros livres a serem configurados, há uma limitação com relação ao seu desempenho. Quando comparado ao GEO canônico, é evidente o ganho de desempenho e de usabilidade que pode ser obtido pelo A-GEO. Porém,

quando comparado a outros algoritmos evolutivos recentes, o A-GEO possui um desempenho inferior para um conjunto de funções teste com variáveis contínuas (BARROCA, 2019). Isso pode se dar em razão da utilização de codificação binária, visto que a precisão das variáveis de projeto é sempre definida em função da quantidade de bits que a codificam, ou seja, sempre é necessário estabelecer a resolução da busca. Isso pode fazer com que o algoritmo seja incapaz de encontrar o mínimo global, se este não pertencer ao conjunto de soluções possíveis de serem testadas.

O A-GEO implementou grandes melhorias e, salvo engano, é a primeira versão adaptativa do GEO, um algoritmo apresentado há quase duas décadas. A melhoria do A-GEO tem como objetivo uma nova versão adaptativa de estado da arte e competitiva com algoritmos de otimização de alto desempenho, levando em consideração que diversos problemas de engenharia possuem variáveis contínuas.

1.1 Objetivos do trabalho

O objetivo geral do trabalho é o aumento do desempenho e aplicabilidade do algoritmo A-GEO. Para isso, a avaliação do desempenho das novas implementações do A-GEO se dará por meio da utilização de funções teste e, posteriormente, de um conjunto maior de funções, onde os algoritmos foram comparados com outro algoritmo evolutivo. Espera-se como resultado uma nova versão do A-GEO amigável ao usuário, sem a necessidade de ajuste de parâmetros e com alto desempenho, sendo esses elementos altamente desejados para serem utilizados na busca pela automatização da criação de soluções em projetos complexos, como o projeto conceitual de sistemas espaciais. Após a fase de aprimoramento, o algoritmo será utilizado para solucionar um problema de otimização na fase conceitual de uma missão espacial, tendo o intuito de demonstrar a aplicabilidade do algoritmo desenvolvido em centros de projeto integrado, como o Centro de Projetos Integrado de Missões Espaciais (CPRIME) da Divisão de Sistemas Espaciais (DISEP) do INPE.

1.1.1 Objetivos específicos

No presente trabalho de pesquisa, para o aumento do desempenho do A-GEO, buscou-se:

- Implementar o mecanismo de controle de parâmetros do A-GEO no GEOvar, uma variante do GEO, verificando essa implementação através de experimentos numéricos utilizando um conjunto de funções teste;
- alterar a codificação das variáveis de projeto do A-GEO de binária para real, não sendo mais necessária a definição da quantidade de bits que codifica cada variável;
- estudar o modo como as variáveis de projeto são perturbadas na codificação real, bem como a forma de realizar diversas mutações em cada variável em uma única iteração;
- investigar o mecanismo de controle do parâmetro τ proposto para o A-GEO, o que pode indicar outras maneiras de controlar o parâmetro durante a busca;
- explorar maneiras de tornar o A-GEOreal completamente adaptativo, sem a necessidade de ajuste de parâmetros;
- aplicar o algoritmo aprimorado na busca pela automatização da criação de soluções em um projeto conceitual de sistemas espaciais.

Este trabalho está organizado da seguinte maneira: O Capítulo 2 apresenta uma breve revisão bibliográfica embasando os métodos utilizados, o Capítulo 3 apresenta os materiais e métodos utilizados, o Capítulo 4 apresenta as funções teste utilizadas e a metodologia para os experimentos numéricos, o Capítulo 5 apresenta os resultados e as discussões dos resultados experimentais, o Capítulo 6 mostra os desempenhos utilizando um conjunto de funções teste do CEC 2017 e o Capítulo 7 aborda sobre a aplicação dos algoritmos em uma otimização de sistemas espaciais.

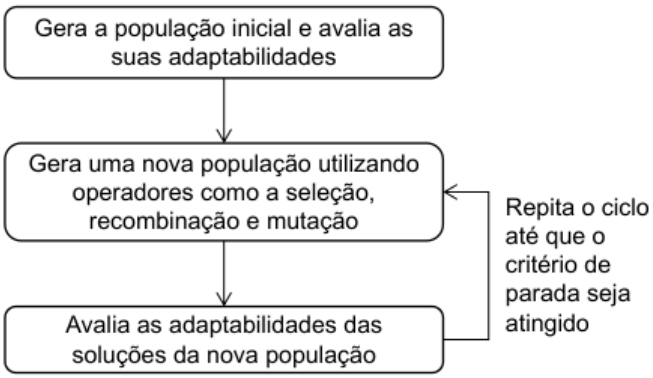
2 REVISÃO BIBLIOGRÁFICA

2.1 Algoritmos evolutivos

Dentre os algoritmos cada vez mais populares na computação inteligente e amplamente aplicados à um grande número de problemas estão as meta-heurísticas (MAIER et al., 2019; BOUSSAID et al., 2013). As meta-heurísticas podem ser divididas em duas classes: as que trabalham e as que não trabalham com populações de soluções (MAIER et al., 2019). Uma grande e a mais popular classe das meta-heurísticas são os algoritmos evolutivos (AEs), que são abordagens estocásticas baseadas em população (EIBEN; SMITH, 2015). Os AEs são conhecidos por usarem técnicas que imitam a evolução natural e são utilizados em muitas áreas de otimização em ciência e engenharia (EIBEN; SMITH, 2015; SLOWIK; KWASNICKA, 2020).

Muitos algoritmos inspirados na evolução natural mantêm a diversidade de uma população imitando características como reprodução, mutação, seleção, eliminação e migração (DE FALCO et al., 2012). Esses algoritmos inicializam aleatoriamente uma população e avaliam a qualidade individual utilizando uma função de aptidão. Os indivíduos então exploram o espaço de busca. Se um indivíduo ou seus vizinhos encontrarem uma boa região, essa região será intensamente investigada. Caso contrário, eles ainda exploram extensivamente outras regiões na busca do ótimo global (ZHAO et al., 2019). Um fluxograma genérico dos algoritmos evolutivos é apresentado na Figura 2.1.

Figura 2.1 – Pseudo-código de algoritmos evolutivos.



Fonte: Adaptado de Maier (2019).

Ao explorar o espaço de busca e gerar novas soluções, ao longo das gerações (iterações), a população de soluções é evoluída até que o indivíduo mais apto da população atinja uma métrica de desempenho desejada (KALITA; THANGAVELAUTHAM, 2020). Essa abordagem pode ser muito útil para uma equipe de projeto tomar decisões informadas sobre o projeto do sistema, encontrando um conjunto de soluções quase ótimas que representam as compensações entre objetivos conflitantes, como massa, custo e desempenho (KALITA; THANGAVELAUTHAM, 2020).

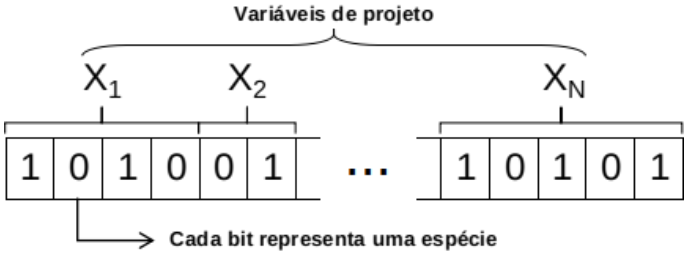
Os algoritmos evolutivos provaram ser altamente eficazes na resolução de uma vasta gama de problemas, uma vez que podem ser utilizados em contextos de decisão (MAIER, 2019). Os AEs podem ser utilizados como um elemento em processos de tomada de decisão, podem ser ligados a modelos de simulação existentes para ajudar na exploração de grandes espaços de solução, podem atender a múltiplos objetivos concorrentes e também podem levar em conta incertezas (MAIER, 2019). Dentre as desvantagens, em seu trabalho, Maier et al. (2014) apresentam que eles são potencialmente dispendiosos em termos computacionais, principalmente dependendo da eficiência computacional do modelo de simulação, que não lhes é garantido identificar a solução globalmente ótima (de uma perspectiva matemática) e que geralmente precisam ser ajustados com o problema em consideração.

2.2 Algoritmo de Otimização Extrema Generalizada (GEO)

O GEO (DE SOUSA et al., 2003) é uma meta-heurística de busca global baseada no método EO proposto por Boettcher e Percus (2001). Ele foi proposto para superar uma dificuldade do método da EO, onde “Uma definição geral para o índice de adaptabilidade das variáveis pode ser ambígua, ou mesmo impossível” (BOETTCHER; PERCUS, 2001). O GEO pode ser aplicado a problemas com espaços de projeto não convexos ou desconexos e com variáveis contínuas, discretas ou inteiras, enquanto possui apenas um parâmetro livre a ser ajustado (DE SOUSA et al, 2003).

De maneira similar a um cromossomo binário em um algoritmo genético (AG), no GEO as variáveis de projeto são codificadas por uma sequência de bits. Todavia, diferentemente do AG, no GEO não há uma população de strings, mas a população de espécies é formada pela sequência (string) de bits, onde cada bit representa uma espécie (DE SOUSA et al., 2003), como apresentado na Figura 2.2.

Figura 2.2 – N variáveis de projeto codificadas em uma sequência binária.



No exemplo, as variáveis de projeto estão codificadas com diferentes tamanhos de string de bits.

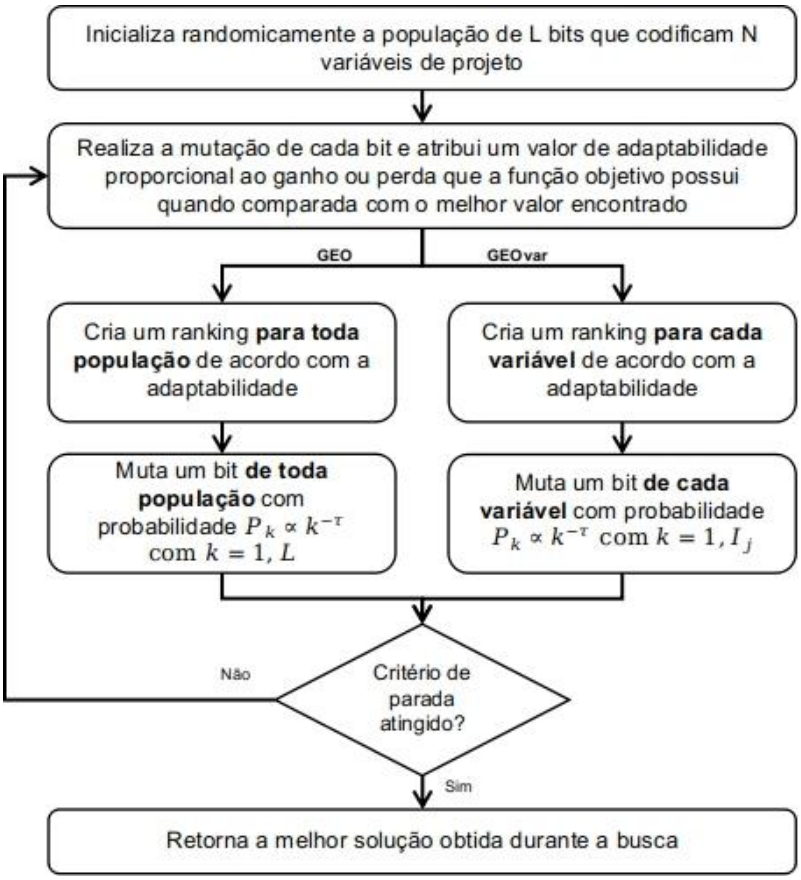
Fonte: Da Luz et al. (2021).

A cada iteração, é avaliado o ganho ou perda no valor da função objetivo quando cada um dos bits é mutado. Esses bits então são ranqueados desde o bit menos adaptado (k = 1; aquele que se mutado representa o maior ganho no valor da função objetivo) até o bit mais adaptado (k = L; aquele que se mutado representa o menor ganho no valor da função objetivo), onde k significa a

posição do ranking e L significa o número total de bits da população. Posteriormente, apenas um bit é escolhido para mutar de forma aleatória. Um bit é confirmado para mutar caso um número aleatório gerado com distribuição uniforme entre o intervalo [0,1] for menor ou igual a $Pi(k) = k^{-\tau}$. Caso contrário, um novo bit é escolhido e o processo é repetido até um bit ser confirmado. Portanto, quanto maior o valor de τ, menor é a probabilidade de um bit com $k > 1$ ser confirmado para mutar, ou seja, mais determinística a busca se torna. Por outro lado, quanto menor o valor de τ, maiores as chances que um bit com $k > 1$, quando escolhido, seja confirmado para mutar, tornando a busca mais estocástica.

A versão denominada GEOvar possui um mecanismo de ranqueamento e mutação diferente do GEO. Enquanto no GEO o ranqueamento é realizado na população inteira e apenas 1 bit é escolhido para mutar a cada geração, no GEOvar os bits são ordenados por variável de projeto e, a cada iteração, um bit de cada uma delas é escolhido para mutar (DE SOUSA et al., 2003). Ou seja, são mutados N bits a cada iteração, onde N é o número de variáveis de projeto. A Figura 2.3 apresenta um diagrama diferenciando as duas implementações.

Figura 2.3 – Diagrama dos algoritmos GEO e GEOvar.



Na figura, L é o número de bits de cada variável, com j = 1, N.

Fonte: Adaptada de De Sousa et al. (2003).

Ao possuir apenas um parâmetro livre para ajustar, o GEO tem uma vantagem a priori, em relação a outros algoritmos como o AG e o Recozimento Simulado (SA – Simulated Annealing), que nas suas formas originais possuem pelo menos três parâmetros livres. Menos parâmetros livres para ajustar significa um custo computacional total menor na utilização do algoritmo, além de torná-lo mais fácil de usar por usuários potenciais.

2.3 Controle de parâmetros

Um aspecto que comumente possui grande influência sobre o desempenho dos AEs para uma certa aplicação é a configuração adequada de seus parâmetros

livres (KARAFOTIAS et al., 2014). O ajuste de parâmetro pode aumentar muito o desempenho dos AEs. Porém, os valores dos parâmetros são especificados antes da execução do AE e estes permanecem imutáveis durante a busca. Segundo Eiben e Smith (2015a), esse uso de parâmetros rígidos está em contraste com o fato de que a execução de um AE é um processo adaptativo intrinsecamente dinâmico.

No fim da década de 90, o trabalho de Eiben et al. (1999) propôs uma unificação para as técnicas de controle de parâmetros, classificando-as de forma clara em estratégias determinísticas, adaptativas e auto adaptativas.

A estratégia determinística tem como mecanismo a alteração dos parâmetros livres com base em um determinado progresso, como por exemplo o tempo de execução ou o número de iterações. Eiben et al. (1999) apresentam que essa abordagem não possui nenhuma noção do progresso atual ao resolver o problema, não havendo um feedback sobre a busca. Essa característica é resolvida através da estratégia adaptativa, na qual os parâmetros livres são alterados com base no progresso da busca. Portanto, os parâmetros livres são modificados com base na informação de que as soluções estão piorando ou melhorando durante a busca (ALETI; MOSER, 2016). Ou seja, o feedback da população é levado em consideração na alteração dos parâmetros livres. Por fim, há as estratégias auto adaptativas, onde os parâmetros livres do algoritmo são codificados na própria população, de modo que estes são evoluídos juntamente ao longo da busca (KARAFOTIAS et al., 2014).

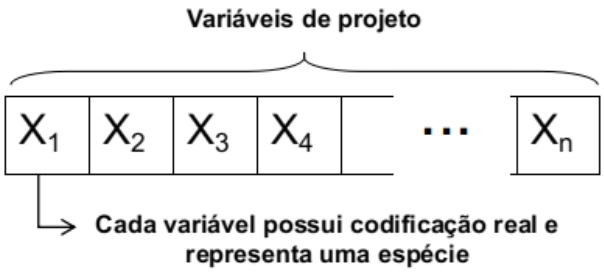
Segundo Eiben e Smith (2015a), foi teoricamente e empiricamente demonstrado em muitas ocasiões que diferentes valores de parâmetros podem ser ótimos em diferentes estágios do processo evolutivo. Diante disso, por exemplo, o espaço de busca pode ser explorado nas primeiras gerações através da maior utilização de mutação, enquanto nas últimas gerações pode ser necessário um ajuste fino de soluções candidatas, reduzindo-se a chance de mutação. Isso implica que o uso de parâmetros estáticos pode levar a um desempenho inferior do algoritmo.

O trabalho de Eiben et al. (1999) observa que encontrar uma função p(t) que altera um dado parâmetro livre ao longo das iterações gera melhores resultados do que manter o valor do parâmetro constante, de forma que a busca seja sintonizada ao longo do tempo.

2.4 GEO com codificação real (GEOreal)

Diferentemente do GEO canônico e suas variantes que possuem a codificação binária das variáveis de projeto, o GEOreal codifica cada variável de projeto com um número real (Figura 2.4). Para um problema com variáveis contínuas, a utilização da codificação binária requer a definição da quantidade de bits que codifica cada variável, ou seja, sempre é necessário estabelecer a resolução da busca. Isso pode fazer com que o algoritmo seja incapaz de encontrar o mínimo global, se este não pertencer ao conjunto de soluções possíveis de serem testadas. Apesar disso, cabe ressaltar-se que a utilização da codificação binária apresenta a vantagem dela incorporar automaticamente as restrições laterais de um problema de otimização, e para uma discretização apropriada das variáveis de projeto, poder apresentar resultados eficientes.

Figura 2.4. N variáveis de projeto com codificação real.



Fonte: Produção do autor.

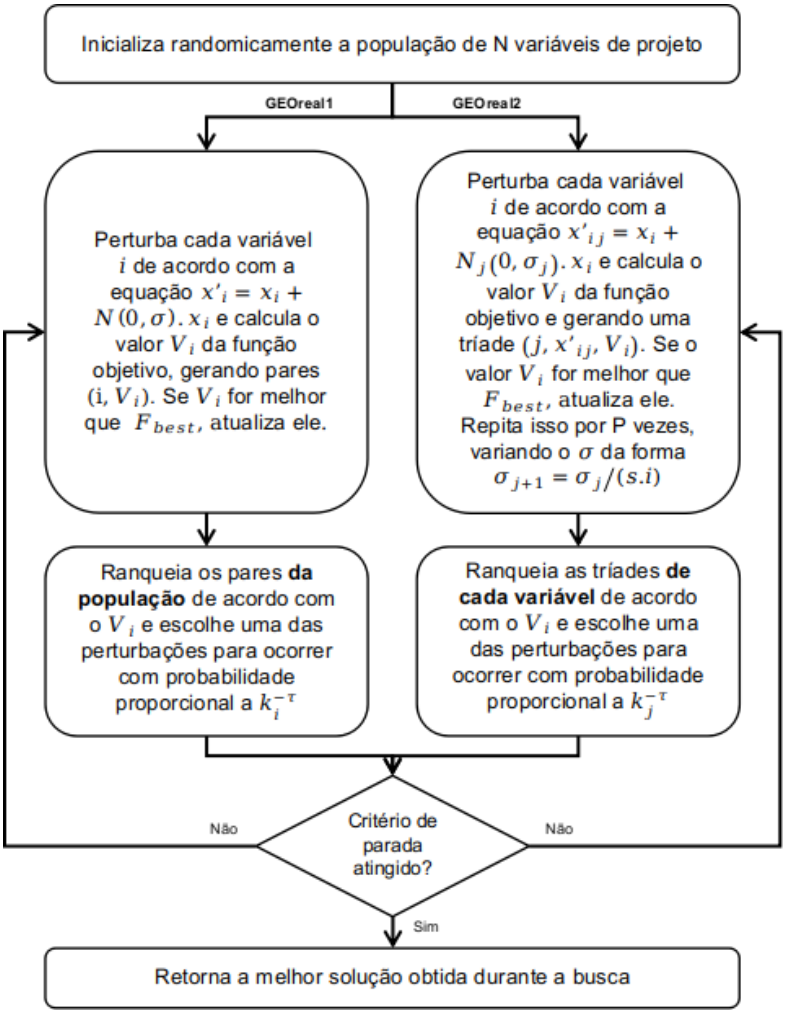
Originalmente, foram desenvolvidas duas versões do GEOreal, denominadas GEOreal1 e GEOreal2. A implementação GEOreal1 possui o mesmo funcionamento do GEO canônico, com exceção da forma como as variáveis de projeto são codificadas e alteradas. Enquanto no GEO canônico a alteração na

variável de projeto é realizada através da mutação de um bit, no GEOreal a mudança na variável de projeto (x_i) é feita através da equação:

$$x'_i = x_i + N(0, \sigma)x_i \quad (2.1)$$

A outra versão do GEOreal denominada GEOreal2 possui origem no GEOvar e tem como diferença em relação ao GEOreal1 a confirmação de uma mutação por variável de projeto a cada iteração. Para isso, são realizadas P mutações em cada variável de projeto e apenas uma delas é confirmada. Mutações na variável que ultrapassam as restrições laterais são penalizadas através da utilização de penalidade externa. O diagrama das duas versões é apresentado na Figura 2.5.

Figura 2.5. Diagrama de fluxo do GEOreal1 e GEOreal2.



Fonte: Adaptado de Mainenti-Lopes et al. (2012).

Uma desvantagem apresentada pelo GEO com codificação real é o aumento no número de parâmetros livres. Além do parâmetro τ que representa o grau de estocasticidade da busca, ambas versões do GEOreal adicionam o parâmetro σ que define o tamanho do passo da mutação no momento de perturbar cada variável de projeto. No caso do GEOreal2, ainda há outros P+1 parâmetros livres. No desenvolvimento original do GEOreal2, Mainenti-Lopes et al. (2012) propuseram uma abordagem para reduzir o número de parâmetros livres. Para isso, ao invés de definir P diferentes desvios-padrões, a abordagem consistiu em gerar esses P desvios-padrões através da regra:

$$\sigma_{j+1} = \frac{\sigma_j}{s \cdot j}, \tag{2.2}$$

onde o desvio padrão da próxima mutação j (σ_{j+1}) é definido como o desvio padrão da mutação atual (σ_j) dividido pelo número da mutação multiplicado por um parâmetro s, onde j = 1...P. É possível observar que quanto maior o valor de s, menor será o desvio padrão da próxima mutação. Além disso, o desvio padrão da primeira iteração de cada variável é dado por σ1, sendo este um dos parâmetros livres do algoritmo. Deste modo, com essa abordagem, os parâmetros livres do GEOreal2 que antes eram τ, σ1 e P+1 parâmetros passaram a ser τ, σ1, P e s. O parâmetro P determina a quantidade de mutações com diferentes desvios-padrões a serem realizadas em cada variável a cada iteração e o parâmetro s significa uma constante que altera o desvio padrão a cada uma das P mutações. Ao utilizar diferentes valores para o desvio padrão para perturbar uma mesma variável a cada iteração, o custo computacional torna-se aproximadamente P vezes maior em comparação ao GEOreal1. Porém, a intenção é fazer com que o algoritmo seja capaz de fazer buscas próximas e distantes do valor da variável em uma única iteração.

2.5 GEO Adaptativo (A-GEO)

Uma versão do GEO que implementa uma técnica de controle de parâmetro adaptativo foi denominada A-GEO (BARROCA, 2019), tendo como objetivo alterar o valor do parâmetro livre τ do GEO automaticamente durante a busca,

equilibrando a capacidade de exploração e aprimoramento (exploration vs. exploitation). Enquanto a exploração permite o algoritmo realizar uma busca ampla por soluções no espaço de projeto, o aprimoramento privilegia a realização de uma busca local (ČREPINŠEK et al., 2013).

A métrica utilizada para medir o melhoramento de desempenho durante a execução do A-GEO foi denominada chance de melhoramento (CoI - Chance of Improvement) e é calculada a cada iteração i através da divisão do número de bits que quando mutados provém valores melhores que um valor de referência ($L_{sucesso}$), dividido pelo número total de bits da população (L), conforme demonstrado em:

$$CoI_i = L_{sucesso} / L. \tag{2.3}$$

Por exemplo, se uma população de 20 bits possui 5 bits que quando mutados provém uma solução melhor que a de referência, então a CoI vale 0,25. O mecanismo de controle do parâmetro τ com base na busca é mostrado na Equação 2.4, onde as condições de controle do parâmetro são apresentadas.

$$\begin{cases} \mathrm{CoI_i} = 0, \text{reinicia o } \tau \\ \mathrm{CoI_i} \leq \mathrm{CoI_{i-1}}, \text{aumenta o } \tau \\ \text{Caso contrário, } \tau \text{ se mantém} \end{cases} \tag{2.4}$$

Quando a população da geração atual não possui maneiras de melhorar em relação à população de referência, ocorre a reinicialização do τ, possibilitando a busca ficar mais estocástica. Assim, reduz-se a chance de esta ficar presa em mínimos locais, permitindo a exploração de mais pontos no espaço de projeto. Já quando essa população possui alguma chance de melhorar em relação a última iteração, ocorre o aumento do τ, tornando a busca mais determinística. Caso nenhuma dessas condições sejam atendidas, não há alteração no valor de τ. A equação aplicada na reinicialização do parâmetro τ durante a busca é:

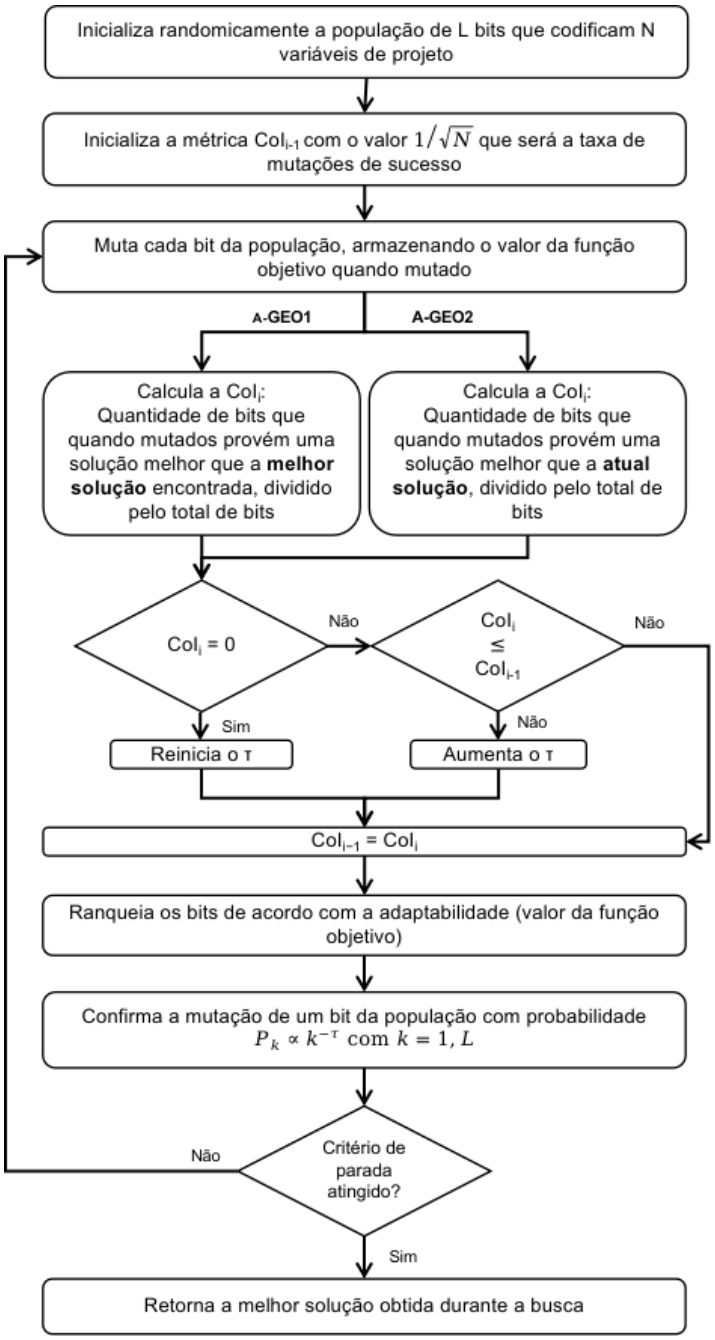
$$\tau = 0.5 \times \text{lognormal}(0, 1/\sqrt{N}), \tag{2.5}$$

e a equação aplicada no aumento do parâmetro τ durante a busca é:

$$\tau = \tau + (0.5 + CoI_i) \times U(0,1). \tag{2.6}$$

O A-GEO foi desenvolvido em duas versões, denominadas A-GEO1 e A-GEO2. A diferença se dá no valor de referência utilizado para o cálculo do Col_i. No A-GEO1, o valor de referência é o melhor valor da função objetivo encontrada em toda a busca até a iteração presente, enquanto na segunda o valor de referência é o valor da função objetivo da população corrente (BARROCA, 2019). Isso significa que no A-GEO1 o τ só reinicia quando nenhum dos bits mutados melhora a solução em relação a melhor já encontrada até o momento na busca. No A-GEO2, o τ reinicia sempre que nenhum dos bits mutados melhora a solução em relação a população atual. A Figura 2.6 apresenta um diagrama de fluxo das duas versões.

Figura 2.6. Diagrama de fluxo dos algoritmos A-GEO1 e A-GEO2.



Fonte: Adaptado de Barroca (2019).

2.6 Otimização no projeto de sistemas espaciais

O projeto de sistemas espaciais é uma atividade complexa, que envolve diversas áreas do conhecimento (WERTZ et al., 2011). Estas características podem levar a exploração limitada de soluções, e ser custosa em tempo, se

esta tarefa depender apenas do conhecimento acumulado do time de projeto. Neste contexto, a otimização numérica mostra-se como uma ferramenta muito apropriada para a geração automática de soluções, em auxílio ao time de especialistas.

Formulado como um problema de otimização, mono ou multiobjetivo, o espaço de projeto pode ser explorado de forma muito mais ampla do que seria, se fosse explorado apenas “manualmente” pela equipe de especialistas. De fato, devido à complexidade encontrada nestes tipos de problema, usualmente termina-se a atividade de projeto assim que seja encontrada uma solução que atenda aos requisitos colocados para o sistema. O uso da otimização numérica permite a automatização da busca por soluções, e assim muitas alternativas podem ser testadas, na busca daquela que represente a melhor solução (ou soluções) para o projeto.

A otimização numérica pode ser aplicada para diversas tarefas no projeto de sistemas espaciais, ou partes do mesmo. Por exemplo, no trabalho de Muraoka et al. (2006) lidou-se com um problema de projeto térmico de uma nave espacial, utilizando o algoritmo de otimização GEO para minimar o consumo de energia dos aquecedores e maximizar as margens de temperatura. Os autores concluíram que a configuração ótima foi rapidamente determinada com um baixo custo computacional e de análise e que a utilização de um algoritmo de otimização para resolver um problema como este é especialmente útil para a análise de plataformas multimissão, que são projetadas para diferentes condições orbitais e modos operacionais. Cuco et al. (2015) buscou resolver um problema de distribuição de equipamentos eletrônicos nos painéis estruturais por meio do uso de otimização. Para isso, os autores utilizaram uma abordagem multiobjetiva para resolver o problema de layout tridimensional, levando em consideração parâmetros como massa, inércia, temperatura e requisitos dos subsistemas. Meziane-Tani et al. (2016) implementaram um AG para projetar uma constelação de satélites dedicados a fornecer uma cobertura mútua contínua da rede sismológica do norte da Argélia. Eles usaram um AG multiobjetivo, denominado como Multi-objecitive Genetic Algorithm (MOGA), visando reduzir o tamanho da constelação e a altitude do satélite. O uso do

algoritmo levou os autores a propor duas soluções ótimas consideradas satisfatórias quando comparadas com um desenho geométrico tradicional (36 satélite utilizando MOGA contra 60 satélites utilizando desenho geométrico tradicional). Outro trabalho que utilizou AG foi o de Paek, Kim e Weck (2019) onde os autores aplicaram técnicas de otimização de recozimento simulado (SA - Simulated Annealing) e algoritmo genético em projeto de constelação de satélite reconfigurável (ReCon - Reconfigurable satellite Constellation). Eles verificaram que tanto o SA quanto o AG produziram soluções ótimas muito semelhantes e que a qualidade da solução AG foi ligeiramente melhor. Além disso, a solução encontrada para o satélite reconfigurável ofereceu um custo benefício superior quando comparado a constelações estáticas. Em seu trabalho, Kalita e Thangavelautham (2020) utilizaram um AE para o projeto de um CubeSat, fazendo uma analogia a organização de uma mochila, cujo objetivo é empacotar o máximo de itens possíveis e não exceder a capacidade da mochila. Ou seja, o objetivo foi empacotar efetivamente os componentes e subsistemas dentro de uma restrição de massa e volume especificada, para que a espaçonave atendesse aos objetivos da missão definidos. Os autores obtiveram resultados satisfatórios na busca de projetos quase ótimos, concluindo que AEs tem o potencial de produzir projetos que maximizem uma função de custo específica da missão, minimizando o custo e a massa. Outro trabalho que também utilizou um AG como base foi o de Gentile et al. (2020), que buscou aplicar o algoritmo de otimização Structured-Chromosome Genetic Algorithm (SCGA) para resolver um problema de design preliminar de um sistema espacial. Os autores concluíram que os resultados foram bons devido ao algoritmo ser capaz de trabalhar com espaços de busca dinâmicos e com diversos tipos de variáveis.

O uso de otimização numérica no projeto de sistemas espaciais é particularmente útil durante a fase conceitual da missão, onde o leque de opções é mais amplo, e onde os maiores impactos no futuro desempenho e custo do sistema são maiores.

3 METODOLOGIA E DESENVOLVIMENTO DE NOVAS IMPLEMENTAÇÕES DO GEO ADAPTATIVO

A metodologia utilizada para o desenvolvimento da dissertação desdobrou-se nas seguintes etapas principais:

a) Revisão bibliográfica sobre o Algoritmo da Otimização Extrema Generalizada (GEO) e suas variações;

b) revisão bibliográfica sobre Algoritmos Evolutivos e Controle de Parâmetros;

c) familiarização e implementação numérica e dos algoritmos GEO, GEOvar e A-GEO;

d) implementação e testes de uma versão adaptativa para o GEOvar (A-GEOvar);

e) implementação do GEO com codificação real e estudo de diferentes formas de alterar as variáveis de projeto e de perturbar diversas vezes as variáveis em uma única iteração;

f) implementação e testes de versões adaptativas para o GEO com codificação real (A-GEOreal);

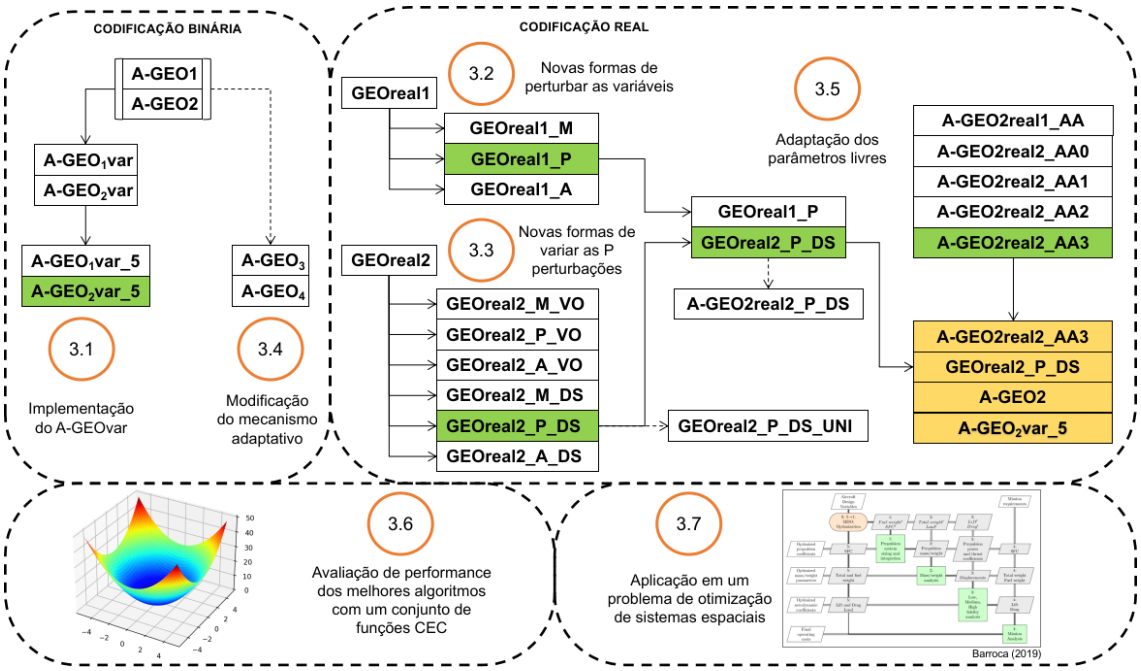
g) investigação de possíveis alternativas para a abordagem de controle do parâmetro τ;

h) aplicação e avaliação de uma ou mais versões adaptativas do GEO desenvolvidas no presente trabalho em um problema de otimização de sistemas espaciais.

A Figura 3.1 apresenta um fluxograma geral das implementações numéricas, que compreendem as etapas “d” até “h” da metodologia. Uma descrição das principais atividades que compreendem essas etapas é apresentada a seguir nas Subseções 3.3 a 3.9. No Capítulo 4, são apresentadas as funções e a metodologia utilizada para os experimentos numéricos. No Capitulo 5, são apresentados os resultados obtidos com a execução dessas etapas para um conjunto de funções teste. No Capítulo 6, serão apresentados os desempenhos

dos melhores algoritmos desenvolvidos para um conjunto de 30 funções teste da competição CEC 2017. No Capítulo 7, os melhores algoritmos serão aplicados em um projeto conceitual de um sistema espacial. Por fim, no Capítulo 8 são apresentadas as conclusões do trabalho e sugestões para a sequência do mesmo.

Figura 3.1 - Fluxograma das implementações numéricas do trabalho.



Os números indicados representam as subseções deste Capítulo 3, onde são descritos os métodos utilizados em cada etapa e os blocos em verde representam os algoritmos que obtiveram o melhor desempenho em cada etapa.

Fonte: Produção do autor.

A primeira etapa, dissertada na Seção 3.3, apresenta a implementação do algoritmo A-GEOvar que possui como base os algoritmos A-GEO e GEOvar. A próxima etapa consistiu em estudar o GEO com codificação real, visando estudar novas maneiras de perturbar as variáveis de projeto no GEOreal1, sendo essa etapa apresentada na Seção 3.4. A seguir, a Seção 3.5 apresenta um estudo do GEOreal2 e de como são realizadas as P mutações nas variáveis de projeto a cada iteração, implementando e avaliando também uma modificação utilizando a distribuição uniforme. A seguir, a Seção 3.6 apresenta

maneiras de modificar o mecanismo adaptativo do A-GEO, e na Seção 3.7 foram desenvolvidas versões semi-auto-adaptativas do GEO com codificação real e um comparativo entre as melhores versões desenvolvidas em todo este trabalho foi apresentado. Por fim, a Seção 3.8 apresenta a abordagem utilizada para avaliação do desempenho dos algoritmos utilizando um conjunto de funções do 2017 IEEE Congress on Evolutionary Computation 2017 (IEEE CEC2017) e a Seção 3.9 introduz a abordagem utilizada na aplicação dos melhores algoritmos adaptativos em um problema de otimização de sistemas espaciais.

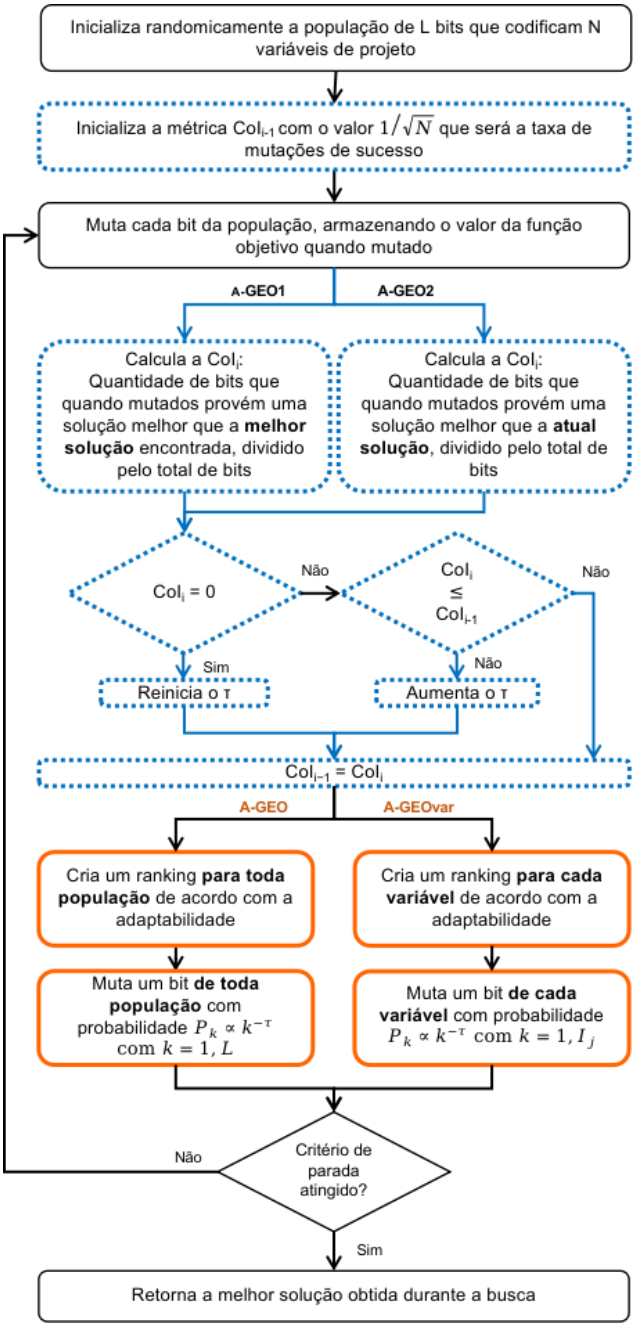
3.1 Implementação do A-GEOvar

O GEOvar mostrou-se mais eficiente do que o GEO na busca pelo ótimo em espaços de projeto com funções contínuas e restrições laterais (DE SOUSA et al., 2003). Desta forma, era de se esperar que uma versão adaptativa daquele também apresentasse um desempenho superior ao A-GEO. Portanto, a primeira implementação realizada, denominada A-GEOvar, visou a melhoria de desempenho do A-GEO através da alteração na maneira como os bits eram ranqueados (ordenados), assim como foi realizado no GEOvar. O mecanismo de controle do parâmetro τ foi o mesmo utilizado para o A-GEO. Para cada iteração, o cálculo da CoI do mecanismo de adaptação, bem como a atualização do valor de τ foram realizados antes do ranqueamento e mutação dos bits. Assim como no caso do A-GEO, duas versões do A-GEOvar foram implementadas, sendo denominadas A-GEO1var e A-GEO2var. Essas versões diferenciam-se nas soluções de referência utilizadas para o cálculo do CoI, sendo elas a melhor solução durante a busca e a solução corrente, respectivamente.

A avaliação de desempenho das implementações foi realizada por meio da utilização de um conjunto de funções teste queá será apresentado na Seção 4.1. A Figura 3.2 mostra em azul os blocos do mecanismo adaptativo para A-GEO1 e A-GEO2, e em laranja é apresentado um comparativo entre as

versões A-GEO e A-GEOvar. Com isso, esse diagrama permite representar o fluxograma das versões A-GEO1, A-GEO2, A-GEO1var e A-GEO2var.

Figura 3.2. Diagrama de fluxo do A-GEOvar.



Em azul, situam-se os blocos que fazem parte do mecanismo adaptativo. Em laranja, situam-se os caminhos para o A-GEO e o A-GEOvar.

Fonte: Produção do autor.

3.2 Aumento de desempenho do GEO com codificação real a partir do GEOreal1 utilizando novas formas de perturbar as variáveis de projeto

Apesar do melhor desempenho do A-GEO em relação ao GEO, ambas versões possuem a codificação binária das variáveis de projeto, sendo necessária a definição da quantidade de bits que codifica cada variável. As versões do GEO com codificação real desenvolvidas por Mainenti-Lopes et al. (2012) apresentaram um bom desempenho para o conjunto de funções estudado quando comparado com as implementações com codificação binária. Desta forma, tornou-se interessante o desenvolvimento de uma versão do A-GEO utilizando a codificação real, que permitiria a variação de cada variável de forma contínua. Diante disso, parte deste trabalho visando o aumento do desempenho do A-GEO foi a adaptação do A-GEO para operar diretamente com a codificação real das variáveis de projeto. Para tanto, um estudo de aumento de desempenho do GEOreal1 foi realizado visando testar diferentes formas de perturbar as variáveis de projeto.

O algoritmo GEOreal desenvolvido originalmente por Mainenti-Lopes et al. (2012) foi implementado com um tipo de mutação nas variáveis que utiliza a distribuição normal. Um valor aleatório da distribuição normal com média 0 e desvio-padrão σ é multiplicado pelo valor atual da variável de projeto e isso é somado ao valor corrente da variável. Essa multiplicação do valor aleatório pelo seu valor corrente faz com que o tamanho da mutação seja dependente do valor desta, ocorrendo uma alta alteração da variável quando o valor dela é alto e um passo baixo quando o valor da variável é pequeno.

A multiplicação do valor da variável pela distribuição faz com que a variável possua grande influência na mutação, visto que um alto valor da variável poderá resultar em um grande passo no espaço de projeto e um pequeno valor da variável poderá resultar em um pequeno passo, mesmo que os valores atribuidos à variável aleatória sejam pequenos e grandes, respectivamente. Esse comportamento do GEOreal original foi observado no trabalho de (DA LUZ et al., 2021) e faz com que a característica de aprimoramento da busca

(exploitation) seja enfraquecida, uma vez que a mutação é diretamente dependente do valor atual da variável.

Em razão dessa influência do valor da variável, três novas formas de perturbar as variáveis de projeto foram testadas para as versões com codificação real e são apresentadas na Tabela 3.1. Para os propósitos deste trabalho, essas formas foram denominadas mutação Multiplicativa (M), mutação Porcentagem (P) e mutação Aditiva (A).

Nas mutações Multiplicativa (M) e Aditiva (A), o parâmetro livre σ (desvio padrão) define o tamanho do passo da mutação no espaço de projeto. Um passo grande significa um grande salto no espaço de projeto, característica da exploração, enquanto que um pequeno passo indica um pequeno deslocamento no espaço de projeto, característica da intensificação da busca.

Na mutação Aditiva, um valor da distribuição gaussiana é somado diretamente ao valor atual da variável. Isso permite que grandes ou pequenos passos no espaço de projeto ocorram dependendo apenas do desvio padrão σ. Na mutação Porcentagem, também é utilizada a distribuição gaussiana e a única diferença para a mutação Aditiva é que o parâmetro σ é calculado com base no intervalo de variação da variável e na porcentagem ρ deste intervalo. Portanto, por exemplo, se o intervalo de variação da variável é [-5, +5] e a porcentagem ρ vale 100%, então o valor de σ será 10. Com isso, esse tipo de mutação desenvolvida possui a capacidade de controlar a capacidade entre exploração e intensificação da busca, uma vez que uma porcentagem alta possui mais chance de produzir um maior passo no espaço de projeto (exploração) e uma porcentagem menor possui mais chance de realizar uma alteração pequena na variável (intensificação). A princial característica deste tipo de mutação desenvolvida é que, em qualquer um dos casos, sempre ocorrerá da maioria dos valores a serem amostrados pela distribuição gaussiana estar situada dentro do intervalo de variação da variável. Essa limitação permite que mutações muito grandes não sejam realizadas frequentemente, as quais seriam penalizadas.

Tabela 3.1. Tipos de mutações testadas.

Mutação	Equação	Parâmetros Livres
Multiplicativa (M)	$x_i = x_i + N(0,\sigma) \cdot x_i$ $\{\sigma \in \mathbb{R} \mid 0 < \sigma < \infty\}$	σ
Porcentagem (P)	$x_i = x_i + N(0,\sigma),$ $\sigma = \rho \cdot (x^U - x^L)$ $\{\rho \in \mathbb{R} \mid 0 < \rho \leq 1\}$	ρ
Aditiva (A)	$x_i = x_i + N(0,\sigma)$ $\{\sigma \in \mathbb{R} \mid 0 < \sigma < \infty\}$	σ

Fonte: Produção do autor.

Essa tabela contém a mutação Multiplicativa que foi desenvolvida no GEOreal1 e as outras mutações testadas, denominadas Porcentagem e Aditiva. A mutação Aditiva foi introduzida nas estratégias evolutivas no trabalho de (VENT, 1975) e consiste em apenas um valor da distribuição gaussiana com média 0 e desvio-padrão σ somado à variável de projeto. A mutação porcentagem, que no nosso conhecimento é melhor introduzida neste trabalho, utiliza a mesma distribuição de probabilidade, porém o desvio padrão é dado como uma porcentagem do intervalo de variação das variáveis de projeto.

3.3 Variações da maneira de perturbar as variáveis de projeto a partir do GEOreal2

Um outro passo para a adaptação do A-GEO para operar diretamente com a codificação real das variáveis de projeto é alterar a Equação 2.2 apresentada na Seção 2.4. Essa equação é utilizada no algoritmo GEOreal2 para variar o desvio padrão durante as P mutações, permitindo o algoritmo realizar várias mutações nas variáveis de projeto a cada iteração. Uma nova abordagem é desenvolvida e comparada com a abordagem original visando aumentar o desempenho do GEO com codificação real. Para os propósitos deste trabalho, essa nova abordagem foi denominada Variação Divisão por S (DS),

GEOreal2_DS, e a abordagem original foi denominada Variação Original (VO), GEOreal2_VO. Ambas abordagens são apresentadas na Tabela 3.2.

A proposta para reduzir a quantidade de parâmetros livres foi definir apenas a quantidade P de mutações e utilizar a Equação 2.2 que necessita apenas do desvio padrão inicial σ1 e do parâmetro s, gerando P diferentes desvios padrões. A nova abordagem apresentada na Tabela 3.2 faz com que a diferença entre cada um dos P desvios padrões seja menor e os P desvios padrões possam ser gerados de maneira logaritmica, por exemplo, quando s vale 10.

Tabela 3.2. Maneiras de modificar o σ durante as P mutações.

Variação	**Equação**	**Parâmetros Livres**
Original (VO)	$\sigma_{j+1} = \frac{\sigma_i}{j \cdot s}$	σ
Divisão por s (DS)	$\sigma_{j+1} = \frac{\sigma_i}{s}$	σ

Fonte: Produção do autor.

Nas equações, σ_{j+1} significa o desvio padrão da próxima mutação j, σ_j é definido como o desvio padrão da mutação atual, j é o número da mutação atual, que varia entre 1 e P, e s é um parâmetro livre que influencia em como variam os P desvios padrões. Em ambas abordagens, a capacidade do algoritmo de fazer buscas próximas e distantes do valor da variável em uma única iteração é mantida. Essas duas variações foram implementadas e seus desempenhos foram avaliados juntamente com as três maneiras de alterar as variáveis de projeto que foram apresentadas na Tabela 3.3, gerando 6 versões do algoritmo GEOreal2 (GEOreal2_M_VO, GEOreal2_M_DS, GEOreal2_P_VO, GEOreal2_P_DS, GEOreal2_A_VO e GEOreal2_A_DS).

Isso permite avaliar qual a melhor abordagem para variar os desvios padrões e qual a melhor maneira de perturbar as variáveis de projeto.

3.4 Modificação do mecanismo adaptativo

O parâmetro τ define o grau de estocasticidade do GEO, ou, do ponto de vista de um algoritmo evolutivo, como se dá a dinâmica exploration x exploitation na busca pelo ótimo no espaço de projeto. A partir da maneira sugerida por Barroca (2019) em como alterar o valor de τ, e da análise dos resultados obtidos por aquele autor para o comportamento do valor de τ ao longo da otimização, buscou-se investigar formas alternativas de fazê-lo, de forma a aumentar o desempenho do GEO adaptativo. Duas novas alternativas denominadas A-GEO3 e A-GEO4 foram propostas e são apresentadas Tabela 3.3.

Tabela 3.3. Comparativo da lógica de controle do mecanismo adaptativo das versões A-GEO1 e A-GEO2 com as versões A-GEO3 e A-GEO4.

Algoritmo	Controle do Mecanismo	Descrição
A-GEO1 e A-GEO2	$\begin{cases} \mathrm{CoI_i} = 0, \text{reinicia o } \tau \\ \mathrm{CoI_i} \leq \mathrm{CoI_{i-1}}, \text{aumenta o } \tau \\ \text{Caso contrário, mantém } \tau \end{cases}$	Quando não há mais nenhum indivíduo que melhora a busca, reseta o τ. Quando há menos indivíduos que melhoram em relação a iteração anterior, aumenta o τ. Caso contrário, se mantém.
A-GEO3	$\begin{cases} \mathrm{CoI_i} = 0, \text{reinicia o } \tau \\ \mathrm{CoI_i} \leq \mathrm{CoI_{i-1}}, \text{aumenta o } \tau \\ \mathrm{CoI_i} > \mathrm{CoI_{i-1}}, \text{diminui o } \tau \end{cases}$	Quando não há mais nenhum indivíduo que melhora a busca, reseta o τ. Quando há menos indivíduos que melhoram em relação a iteração anterior, aumenta o τ. Se há mais indivíduos que melhoram a busca em relação a iteração anterior, diminui o τ.
A-GEO4	$\begin{cases} \mathrm{CoI_i} = 0, \text{reinicia o } \tau \\ \mathrm{CoI_i} > 0, \text{aumenta o } \tau \end{cases}$	Quando não há mais nenhum indivíduo que melhora a busca, reseta o τ. Quando há qualquer indivíduo que melhore a busca, aumenta o τ.

Fonte: Produção do autor.

Ambas versões diferem do A-GEO original nas leis de aumento e reinicialização do τ. Nas leis propostas, CoI_i significa o valor da métrica CoI da iteração atual, que mede a taxa de mutações de sucesso, e CoI_{i-1} significa o valor da métrica na iteração anterior. É importante observar que a métrica CoI é uma razão que mede a porcentagem de bits que quando mutados provém um valor melhor que o valor de referência, então essa razão nunca poderá ser um valor negativo. A solução de referência utilizada para as versões A-GEO3 e A-GEO4 foi a solução corrente assim como no A-GEO2, algoritmo este que já havia apresentado um melhor desempenho.

Na lógica original que é implementada nos algoritmos A-GEO1 e A-GEO2, o τ sempre é resetado quando CoI vale 0. Isso significa que nenhum dos bits quando mutados provém um valor melhor que o valor de referência, o que faz o τ resetar para tornar a busca mais estocástica. Essa lógica foi mantida para as versões A-GEO3 e A-GEO4. A diferença dessas novas implementações está em como modificar o τ quando o CoI não vale 0, ou seja, quando há indivíduos que quando mutados provém um valor melhor que a solução de referência. No caso no A-GEO1 e do A-GEO2, o τ só aumenta quando o valor de CoI for menor que o da iteração anterior, o que significa que nessa iteração menos indivíduos melhoraram em relação à iteração anterior, indicando um aprimoramento da busca. Caso contrário, o τ não é modificado. Essa condição de aumentar o τ foi mantida na versão A-GEO3, porém ao invés de não modificar o τ no caso contrário, o valor de τ é diminuído quando o CoI for maior que o da iteração anterior. Para isso, foi utilizada a mesma forma de alterar o τ de quando ele é aumentado, porém aqui diminuindo o valor dele. A versão A-GEO4 simplifica o mecanismo, uma vez que não é necessário conhecer o valor de CoI da iteração anterior. Essa versão alternativa reseta o valor de τ quando não há indivíduos que melhorem a busca e aumenta o valor de τ sempre que há pelo menos um indivíduo que melhore a busca.

Uma outra abordagem foi implementada e, diferentemente do A-GEO1 onde a população de referência é a melhor da busca até o momento, essa consistia em utilizar como população de referência a solução do melhor indivíduo gerado após as mutações.

3.5 Implementação da versão adaptativa do GEOreal (A-GEOreal) e comparação com melhor versão binária

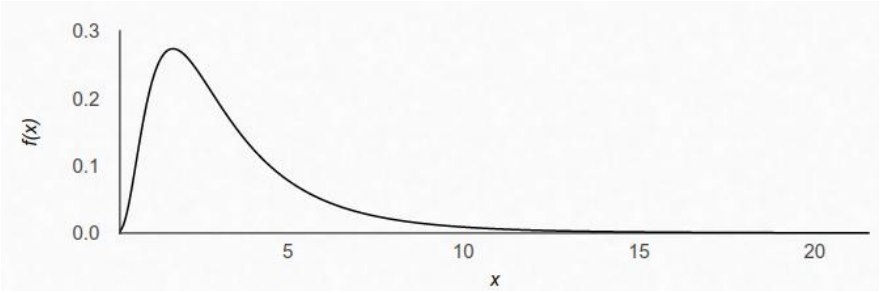
O primeiro experimento visando adaptar o melhor algoritmo obtido na seção anterior consistiu em aplicar o mecanismo de controle adaptativo no parâmetro τ, gerando a versão A-GEO2real2_P_DS. Com isso, o algoritmo com codificação real passou a ter pelo menos um parâmetro livre a menos. Este mecanismo foi o mesmo aplicado no A-GEO2 e utiliza a solução corrente como solução de referência para o cálculo do CoI. A implementação gerada permitiu avaliar diretamente a influência que existe somente na aplicação do mecanismo adaptativo no τ.

Um outro parâmetro livre presente em todas as versões do GEOreal é o desvio padrão σ que é utilizado na mutação das variáveis de projeto. Portanto, diante da influência que esse introduz nos algoritmos com codificação real, um breve estudo sobre a variação deste parâmetro foi realizado. Este parâmetro é relacionado com o ρ nas versões em que a mutação é do tipo Porcentagem (P). Portanto, visando tornar as implementações completamente livre de parâmetros, esse parâmetro foi adaptado utilizando uma estratégia auto-adaptativa. Ela consistiu em codificar este parâmetro juntamente com as variáveis de projeto, atribuir valores de adaptabilidade a ele em cada iteração e ir evoluindo-o de acordo com o feedback da busca. Neste trabalho, a aplicação da estratégia auto-adaptativa foi realizada somente no parâmetro ρ do GEOreal e foram geradas 5 versões ao todo, sendo uma delas para o GEOreal1_P e outras quatro para o GEOreal2_P_DS.

O GEOreal1_P possui como característica perturbar apenas uma vez cada uma das N variáveis de projeto a cada iteração, gerando um valor de adaptabilidade para cada uma delas. Portanto, a fim de gerar a versão auto-adaptativa deste algoritmo, denominada A-GEO2real1_AA, o mecanismo de adaptação foi aplicado para o parâmetro τ e a porcentagem ρ foi adicionada juntamente com as variáveis de projeto. A adaptabilidade de cada variável de projeto foi calculada ao perturbá-la utilizando ρ. Para o cálculo da adaptabilidade da própria porcentagem ρ, foi gerado um ρ’ e este foi utilizado

para perturbar todas as variáveis ao mesmo tempo. No momento de ordenar e escolher uma das alterações para confirmar, todas essas N+1 adaptabilidades foram ordenadas e uma delas foi escolhida para ser confirmada seguindo o mesmo critério do GEO. Como será visto no Capítulo 5, os valores para porcentagem que resultavam no melhor desempenho do GEOreal1_P situavam-se entre 0% e 10%. Portanto, os valores gerados para ρ' durante a busca e o valor inicial de ρ foram gerados através de uma distribuição log-normal com média 1 e desvio padrão 0,67, cuja função densidade de probabilidade é apresentada na Figura 3.3. O valor amostrado na distribuição possui a maioria dos valores entre 0 e 10, favorecendo valores mais baixos, então esse valor amostrado foi dividido por 100 para convertê-lo em uma porcentagem a fim de utilizá-lo nas mutações das variáveis.

Figura 3.3. Função densidade de probabilidade da distribuição log-normal(1, 0.67).

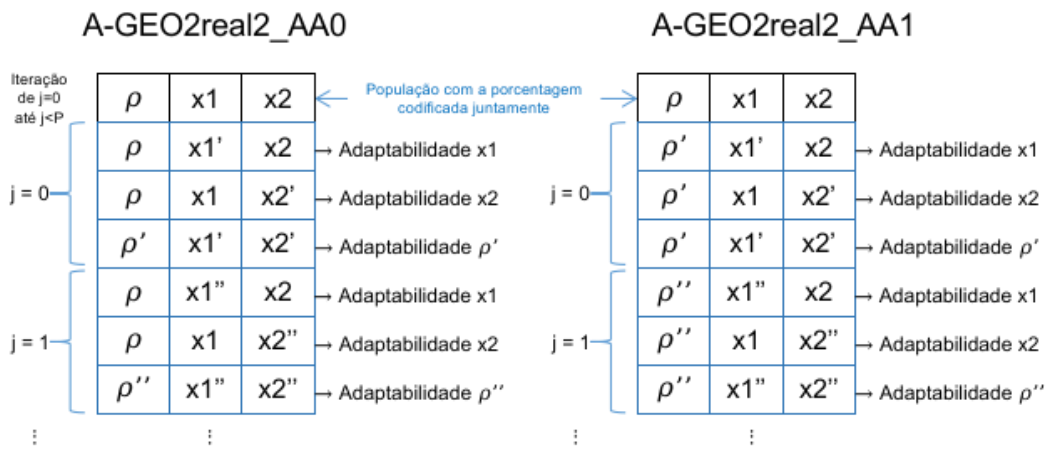


Fonte: Produção do autor.

As versões auto-adaptativas do GEOreal2 foram denominadas A-GEO2real2_AA0, A-GEO2real2_AA1, A-GEO2real2_AA2 e A-GEO2real2_AA3. As duas primeiras versões foram baseadas na versão auto-adaptativa do GEOreal1, ou seja, essas versões utilizaram a mesma distribuição log-normal para gerar valores para ρ', só que a ideia foi realizar múltiplas mutações por variável a cada iteração. O funcionamento da versão AA0 tem como princípio inicializar ρ com um valor da log-normal(1, 0.67) e a cada iteração realizar P mutações nas variáveis de projeto com o mesmo valor de ρ a fim de calcular as adaptabilidades. Para calcular a adaptabilidade de ρ, são gerados P diferentes

ρ’ e estes são utilizados para perturbar todas as variáveis da população ao mesmo tempo e calcular o valor da função objetivo, gerando P adaptabilidades para ρ. Posteriormente, essas P adaptabilidades geradas para cada uma das N+1 variáveis são ordenadas e uma delas é escolhida para ser confirmada, assim como é realizado no GEOreal2. A versão AA1 possui funcionamento semelhante a AA0, com a diferença que ao invés de utilizar o mesmo ρ para perturbar P vezes cada variável de projeto, são utilizados P diferentes ρ’. A Figura 3.4 apresenta um exemplo do cálculo de duas adaptabilidades para cada variável nas versões AA0 e AA1. É possível observar que a única diferença entre ambas versões está nas mutações nas variáveis de projeto da versão AA0 serem realizadas com o mesmo ρ enquanto que na AA1 são gerados P diferentes ρ’.

Figura 3.4. Exemplo do cálculo das adaptabilidades para as versões auto-adaptativas AA0 e AA1.

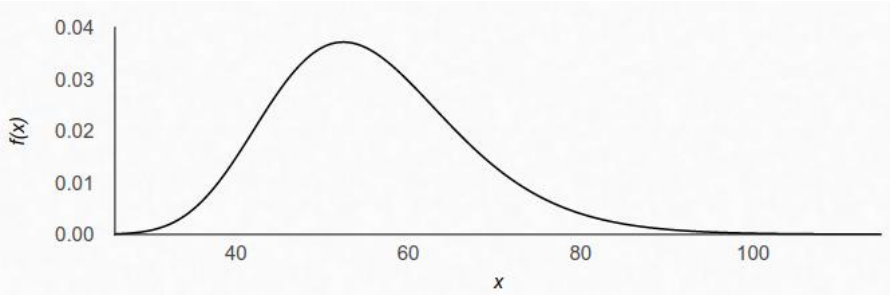


Fonte: Produção do autor.

As versões A-GEO2real2_AA2 e A-GEO2real2_AA3 seguiram a ideia do GEOreal2_P_DS ao utilizar como parâmetro auto-adaptado o ρ1. Diferentemente das implementações AA0 e AA1 onde ocorrem P mutações com diversos ρ’, as versões AA2 e AA3 utilizaram como princípio definir um ρ1 e ir dividindo este por 10 (parâmetro s) durante as 10 (parâmetro P) mutações. O valor para ρ1 analisado na Tabela 3.3 mostrou que são interessantes valores

maiores ou iguais a 10%. Portanto, a geração de novos valores para ρ1 na versão AA2 foi realizada com uma distribuição log-normal de média 4 e desvio padrão 0.2, cuja função densidade de probabilidade é apresentada na Figura 3.5. O valor amostrado na distribuição possui a maioria dos valores entre 10 e 100, então esse valor amostrado foi dividido por 100 para convertê-lo em uma porcentagem.

Figura 3.5. Função densidade de probabilidade da distribuição log-normal(4, 0.2).



Fonte: Produção do autor.

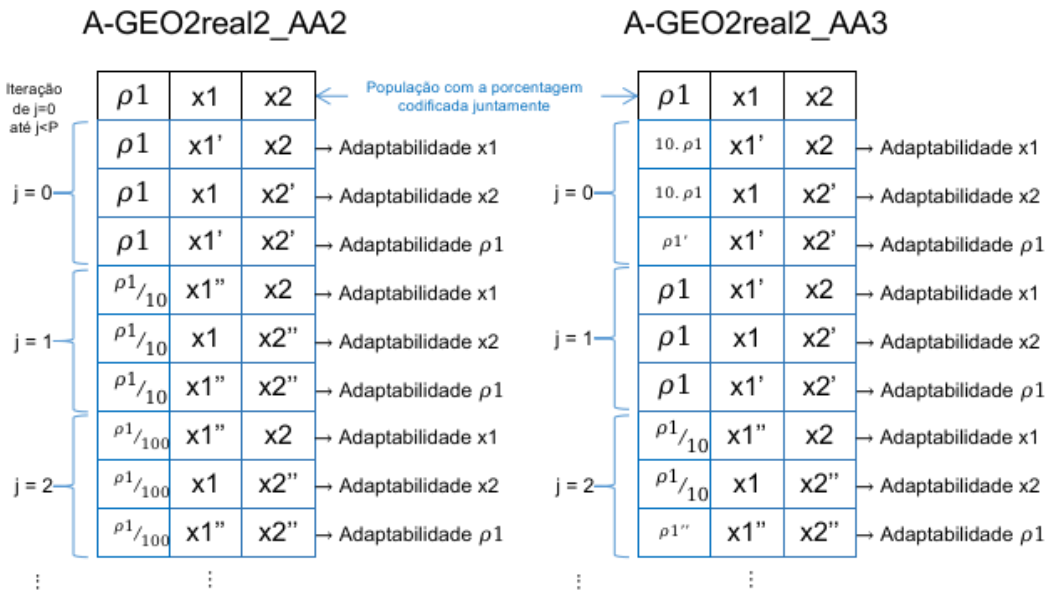
Nesta versão AA2, o valor inicial para ρ é ρ e este vai sendo dividido por 10 em cada uma das 10 mutações. Esses P valores de ρ são utilizados para perturbar cada variável de projeto a fim de calcular a adaptabilidade de cada uma. Para calcular a adaptabilidade de ρ, são perturbadas todas as variáveis de projeto ao mesmo tempo e o valor da função objetivo é calculado. Por fim, todas as adaptabilidades de cada variável de projeto, incluindo as P adaptabilidades de ρ1, são ordenadas de forma crescente e uma delas é aceita, atualizando o valor dessa variável na população corrente para o valor de quando ela foi perturbada. Ou seja, todas as adaptabilidades de ρ1 serão ordenadas e terá maiores chances de ser escolhida para ser aceita aquela cujo ρ resultou na melhor adaptabilidade. Este valor de ρ é o novo ρ1 da próxima geração.

Percebe-se que se a porcentagem inicial ρ1 é um valor entre 0,1 e 1 e esta vai sendo dividida 10 vezes por 10, portanto a última mutação é próxima a 10^{-9} vezes o intervalo de variação da variável. Esse valor é muito pequeno,

representando uma mutação muito pequena na variável. No momento em que um ρ muito pequeno (ex.: 10^{-5}) resulta em uma adaptabilidade muito boa, ele pode ser o novo ρ1 da próxima iteração. Nesta próxima iteração, este ρ1 será dividido novamente 10 vezes por 10, resultando em porcentagens menores que antes, cada vez mais aprimorando a busca. Quando a busca está estocástica (τ foi resetado), o valor de ρ1 também é resetado utilizando a distribuição log-normal apresentada na Figura 3.5, permitindo que grandes mutações ocorram nas variáveis de projeto na etapa de exploração.

Para contornar o fato da porcentagem inicial somente ser dividida várias vezes por 10, atingindo um valor muito pequeno de porcentagem, a versão AA3 propôs partir de uma porcentagem inicial ρ1 pequena e multiplicar ela 4 vezes por 10 e dividir ela 4 vezes por 10, totalizando 9 mutações. Para tanto, foi definido que a porcentagem inicial seja um valor aleatório entre 10^{-4} e 10^{-5}, pois esse valor quando multiplicado 4 vezes por 10 resulta em uma porcentagem entre 10% e 100%. Ou seja, ao invés de gerar um valor alto para a porcentagem inicial e ir dividindo ela várias vezes por 10, essa abordagem gera um valor menor de porcentagem e multiplica e divide ele por 10 para atingir grandes e pequenas porcentagens. Dentre as 9 mutações realizadas em cada uma das N variáveis de projeto, a primeira é feita utilizando ρ1 e as demais são realizadas com 10000ρ1, 1000ρ1, 100ρ1, 10ρ1, ρ1/10, ρ1,100, ρ1/1000 e ρ1/10000. Para calcular as adaptabilidades de ρ1, são gerados 9 diferentes ρ1' entre 10^{-4} e 10^{-5} e a adaptabilidade é obtida ao perturbar todas as variáveis de projeto ao mesmo tempo com este ρ1' e calcular o valor da função objetivo. A Figura 3.6 apresenta um exemplo do cálculo de três adaptabilidades para cada variável nas versões AA2 e AA3. É possível observar que na versão AA2 a porcentagem vai sendo dividida por 10 a cada uma das P mutações, enquanto que na versão AA3 o ρ1 é um valor pequeno que vai sendo multiplicado por 10 em algumas mutações e em outras é dividido por 10. Além disso, sempre o cálculo da adaptabilidade de ρ1 é dado por diferentes ρ1’.

Figura 3.6. Exemplo do cálculo das adaptabilidades para as versões auto-adaptativas AA2 e AA3.



Fonte: Produção do autor.

3.6 Avaliação de desempenho utilizando um conjunto de funções CEC

As melhores versões adaptativas que utilizam codificação real e binária foram utilizadas para a execução de um conjunto de 30 funções com diferentes características topológicas. Essas funções são originadas da competição realizada no 2017 IEEE Congress of Evolutionary Computation (CEC 2017) e contém funções monoobjetivas de diversas classes, como dinâmicas e computacionalmente caras. Um número maior de problemas permitiu uma avaliação mais ampla sobre o desempenho dos algoritmos. O Capítulo 6 deste trabalho dedica-se exclusivamente a abordar e aprofundar este assunto.

3.7 Aplicação em um problema de otimização de sistemas espaciais

Após o desenvolvimento das implementações e do estudo da forma de perturbar as variáveis e melhorar o A-GEO, as implementações foram utilizadas como otimizadores em um problema da fase de projeto conceitual de missão espacial, como por exemplo o abordado por Barroca (2019). O Capítulo 7 deste trabalho é exclusivamente dedicado a este estudo.

3.8 Resumo comparativo entre as diferentes versões desenvolvidas

Como um resumo das várias versões descritas entre as Seções 3.3 e 3.7, a Tabela 3.4 apresenta uma compilação das principais características das implementações.

Tabela 3.4. Comparativo entre todos os algoritmos implementados no estudo.

Algoritmo	**Codificação**	**Maneira de alterar as variáveis**	**Parâmetros Livres com necessidade de tuning**	**Características**
A-GEO1	Binária	Mutação do bit	Nenhum	Apenas uma mutação na população inteira é confirmada. A solução de referência é a melhor solução obtida até o momento da busca.
A-GEO2	Binária	Mutação do bit	Nenhum	Mesmas características do A-GEO1. Porém, a solução de referência é a solução corrente.
A-GEO3	Binária	Mutação do bit	Nenhum	Mesmas características do A-GEO2. Porém, as regras de controle do mecanismo diferem.
A-GEO4	Binária	Mutação do bit	Nenhum	Mesmas características do A-GEO2. Porém, as regras de controle do mecanismo diferem.
A-GEO1var	Binária	Mutação do bit	Nenhum	Mesmas características do A-GEO1. Porém, são mutados um bit por variável.
A-GEO2var	Binária	Mutação do bit	Nenhum	Mesmas características do A-GEO2. Porém, são mutados um bit por variável.
A-GEO2var_5	Binária	Mutação do bit	Nenhum	Mesmas características do A-GEO2var. Porém, o τ é resetado quando ultrapassa o valor 5.
GEOreal1	Real	$xi = xi + xi \cdot N(0, \sigma)$	τ, σ	Após todas variáveis serem perturbadas, apenas uma delas é escolhida para ser confirmada.
GEOreal2	Real	$xi = xi + xi \cdot N(0, \sigma)$	$\tau, \sigma 1, P, s$	Após todas as variáveis serem perturbadas P vezes, uma mutação por variável é escolhida para ser confirmada.
GEOreal1_M	Real	$xi = xi + xi \cdot N(0, \sigma)$	τ, σ	Mesmas características do GEOreal1. A diferença está na maneira de perturbar as variáveis.

(continua)

Tabela 3.4. Continuação.

Algoritmo	Codificação	Maneira de alterar as variáveis	Parâmetros Livres com necessidade de tuning	Características
GEOreal1_P	Real	$xi = xi \cdot N(0, \sigma)$, onde $\sigma = \rho \cdot \Delta xi$	τ, ρ	Mesmas características do GEOreal1. A diferença está na maneira de perturbar as variáveis.
GEOreal1_A	Real	$xi = xi + N(0, \sigma)$	τ, σ	Mesmas características do GEOreal1. A diferença está na maneira de perturbar as variáveis.
GEOreal2_M_VO	Real	$xi = xi + xi \cdot N(0, \sigma)$	$\tau, \sigma 1, P, s$	Mesmas características do GEOreal2. Porém, a maneira de variar o σ durante as P mutações é dada por $\sigma_{j+1} = \sigma_j/(s \cdot i)$.
GEOreal2_P_VO	Real	$xi = xi + N(0, \sigma)$, onde $\sigma = \rho \cdot \Delta xi$	$\tau, \rho 1, P, s$	Mesmas características do GEOreal2_M_VO. A diferença está na maneira de perturbar as variáveis.
GEOreal2_A_VO	Real	$xi = xi + N(0, \sigma)$	$\tau, \sigma 1, P, s$	Mesmas características do GEOreal2_M_VO. A diferença está na maneira de perturbar as variáveis.
GEOreal2_M_DS	Real	$xi = xi + xi \cdot N(0, \sigma)$	$\tau, \sigma 1, P, s$	Mesmas características do GEOreal2. Porém, a maneira de variar o σ durante as P mutações é dada por $\sigma_{j+1} = \sigma_j/s$.
GEOreal2_P_DS	Real	$xi = xi + N(0, \sigma)$, onde $\sigma = \rho \cdot \Delta xi$	$\tau, \rho 1, P, s$	Mesmas características do GEOreal2_M_DS. A diferença está na maneira de perturbar as variáveis.
GEOreal2_A_DS	Real	$xi = xi + N(0, \sigma)$	$\tau, \sigma 1, P, s$	Mesmas características do GEOreal2_M_DS. A diferença está na maneira de perturbar as variáveis.

(continua)

Tabela 3.4. Conclusão.

Algoritmo	Codificação	Maneira de alterar as variáveis	Parâmetros Livres com necessidade de tuning	Características
GEOreal2_A_DS	Real	$xi = xi + N(0, \sigma)$	$\tau, \sigma 1, P, s$	Mesmas características do GEOreal2_M_DS. A diferença está na maneira de perturbar as variáveis.
A-GEO2real2_P_DS	Real	$xi = xi + N(0, \sigma)$, onde $\sigma = \rho \cdot \Delta xi$	$\rho 1, P, s$	Mesmas características do GEOreal2_P_DS. Porém, o parâmetro τ passou a ser adaptado com o mecanismo adaptativo do A-GEO2.
A-GEO2real2_AA0	Real	$xi = xi + N(0, \sigma)$, onde $\sigma = \rho \cdot \Delta xi$	Nenhum	Mesmas características do GEOreal2_P_DS. Porém, os parâmetros P e s foram fixados em 10 e os parâmetros τ e ρ passaram a ser adaptados com os mecanismos adaptativo do A-GEO2 e auto-adaptativo, respectivamente.
A-GEO2real2_AA1	Real	$xi = xi + N(0, \sigma)$, onde $\sigma = \rho \cdot \Delta xi$	Nenhum	Mesmas características do A-GEO2real2_AA0. A diferença está em como o mecanismo auto-adaptativo do parâmetro ρ foi implementado.
A-GEO2real2_AA2	Real	$xi = xi + N(0, \sigma)$, onde $\sigma = \rho \cdot \Delta xi$	Nenhum	Mesmas características do A-GEO2real2_AA0. A diferença está em como o mecanismo auto-adaptativo do parâmetro ρ foi implementado.
A-GEO2real2_AA3	Real	$xi = xi + N(0, \sigma)$, onde $\sigma = \rho \cdot \Delta xi$	Nenhum	Mesmas características do A-GEO2real2_AA0. A diferença está em como o mecanismo auto-adaptativo do parâmetro ρ foi implementado.

Fonte: Produção do autor.

4 PREPARAÇÃO DOS EXPERIMENTOS NUMÉRICOS

Este Capítulo destina-se a apresentar a metodologia utilizada para a realização dos experimentos numéricos do trabalho. Para isso, serão apresentadas as funções teste utilizadas para avaliar o desempenho dos algoritmos desenvolvidos, bem como o ambiente de desenvolvimento utilizado para executar as implementações.

As execuções e os gráficos de desempenho do trabalho a serem descritos podem ser divididos em 3 tipos: ajuste de parâmetros, gráficos demédia de desempenho dos algoritmos e boxplots dos valores atingidos ao fim da busca.

1) **Ajuste de parâmetros:** Alguns algoritmos necessitam da etapa de ajuste de parâmetros a fim de encontrar os parâmetros que resultam no melhor desempenho do algoritmo. Para esse tipo de execução, cada algoritmo foi executado de forma independente 50 vezes com diferentes populações iniciais e o critério de parada definido foi limitado pelo número de avaliações da função objetivo (NFE - Number of Function Evaluations) ou pela precisão do valor final da função. O critério foi de cem mil (100000) NFEs ou uma precisão de 10^{-16} com relação ao f(x)* (valor ótimo conhecido de cada função). Por precisão, o critério de parada é atingido quando, durante a busca, a subtração do melhor valor da função objetivo pelo valor de f(x)* é inferior a precisão de 10^{-16}. A média do melhor valor da função objetivo e a média do NFE atingido ao obter a precisão para as 50 execuções foram obtidas e o melhor conjunto de parâmetros foi utilizado posteriormente para o comparativo de desempenho entre algoritmos.

2) **Gráficos de média de desempenho:** Todas as comparações de desempenho entre algoritmos apresentam o valor médio do melhor f(x) em função do NFE. Para isso, durante a busca foi armazenado o valor do melhor f(x) a cada 500 NFEs e posteriormente foi realizada a média aritmética para cada NFE a fim de gerar os gráficos de desempenho dos algoritmos para cada uma das funções. O critério de parada definido foi de cem mil (100000) NFEs e foram realizadas 50 execuções independentes para cada algoritmo. Cada execução partia de um ponto diferente escolhido aleatóriamente no espaço de

projeto, mas os diferentes algoritmos eram inicializados a partir daqueles mesmos pontos.

Boxplots dos valores atingidos ao fim da busca: Durante as 50 execuções de cada algoritmo partindo de um ponto diferente do espaço de projeto, além de obter o valor a cada 500 NFEs, foi armazenado o melhor valor atingido ao final de cada busca. Estes 50 valores para cada algoritmo serão apresentados após os gráficos de média de desempenho no formato de boxplot. Os boxplots são apresentados na escala logarítmica. O boxplot é uma ferramenta gráfica que permite visualizar a distribuição e os valores discrepantes (outliers) de um conjunto de dados. Isso permite o desenvolvimento de uma perspectiva geral sobre o valor dos dados, uma vez que as medidas de estatísticas descritivas como o mínimo, máximo, 1º quartil, mediana e 3º quartil são apresentados no boxplot. Nele, o retângulo do meio possui três linhas horizontais: a linha de baixo, que indica o 1º quartil, a linha de cima, que indica o 3º quartil, e a linha interna que indica a mediana. A média aritmética será plotada como um círculo na cor preta. Os outliers serão apresentados como círculos na cor vermelha e o limite de detecção deles será de 1,5 vezes o intervalo interquartílico (distância entre o 1º e o 3º quartil). Este intervalo interquartílico é uma estatística mais robusta para medir a variabilidade dos dados, uma vez que ele não sofre influência de outliers, diferentemente da amplitude que é a distância entre o valor máximo e o mínimo.

4.1 Funções teste utilizadas no desenvolvimento dos algoritmos

No presente estudo, a avaliação de desempenho dos algoritmos desenvolvidos foi primeiramente realizada através de um conjunto de 6 funções objetivo. O conjunto escolhido (Tabela 4.1) contém funções de minimização amplamente utilizadas para fins de benchmarking em algoritmos de otimização (ALI et al., 2005; JAMIL; YANG, 2013).

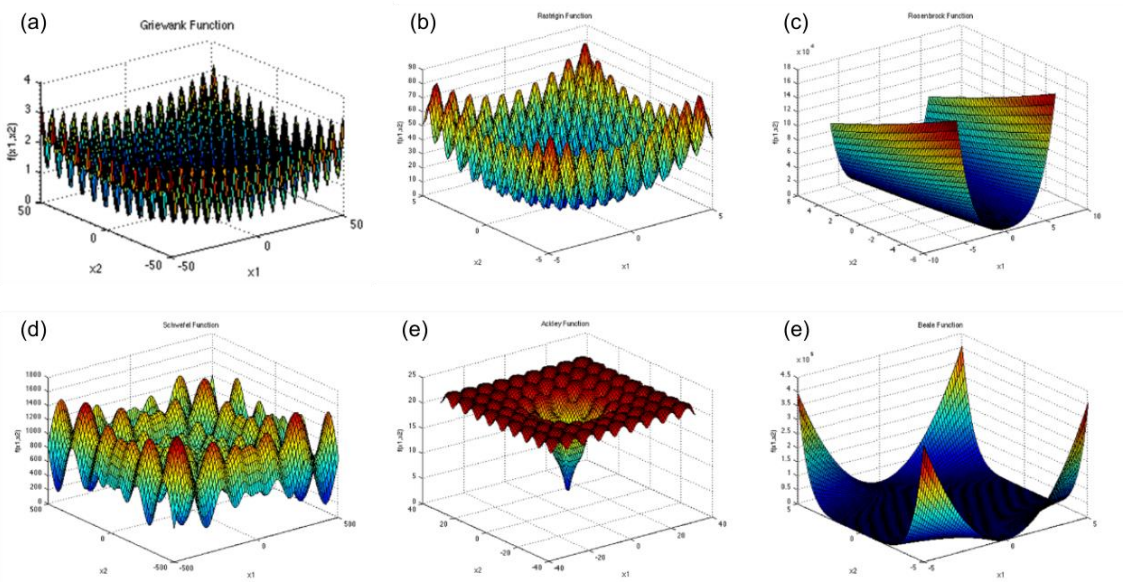
Tabela 4.1. Conjunto de funções teste. A terceira coluna apresenta o número de bits por variável utilizado nas versões binárias do GEO com codificação binária.

Função	Dimensão (N)	Bits por variável	Restrições Laterais	Mínimo global (x*)	f(x*)
Griewangk	10	14	$-600 \leq x_i \leq 600$	(0, 0, ...)	0
Rastrigin	20	16	$-5.12 \leq x_i \leq 5.12$	(0, 0, ...)	0
Rosenbrock	2	13	$-2.048 \leq x_i \leq 2.048$	(1, 1)	0
Schwefel	10	16	$-500 \leq x_i \leq 500$	(420.96, ...)	0
Ackley	30	16	$-30 \leq x_i \leq 30$	(0, 0, ...)	0
Beale	2	10	$-4.5 \leq x_i \leq 4.5$	(3, 0.5)	0

Fonte: Produção do autor.

Este conjunto é composto de funções com diferentes características topológicas (Figura 4.1) e todas as funções descritas a seguir são de minimização. Além disso, a maioria das funções utilizadas são da classe multimodal, sendo apenas a Rosenbrock uma função unimodal.

Figura 4.1 - As representações bidimensionais das funções (a) Griewangk, (b) Rastrigin, (c) Rosenbrock, (d) Schwefel, (e) Ackley e (e) Beale.

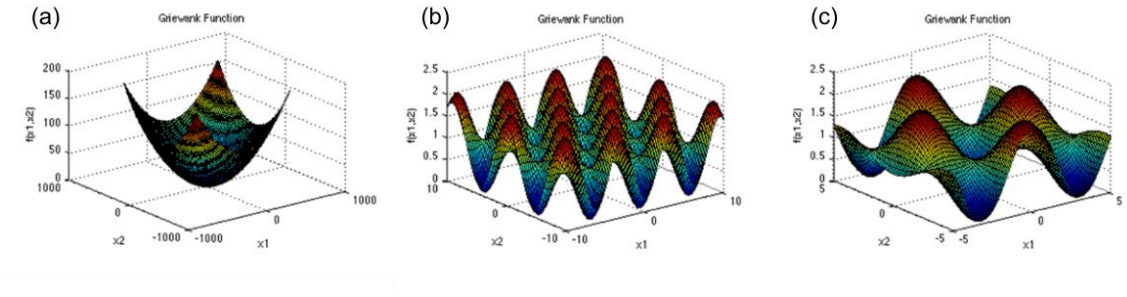


Fonte: Adaptado de Surjanovic e Bingham (2013).

4.1.1 Griewangk

A função Griewangk (Gri) tem sido amplamente usada para testar a convergência das funções de otimização. A interpretação da função muda com a escala, por exemplo, a visão geral sugere uma função convexa (Figura 4.2a), a vista em escala média sugere a existência de diversos mínimos locais (Figura 4.2b) e, finalmente, o zoom nos detalhes indica uma estrutura complexa de vários extremos locais (Figura 4.2c) (MOLGA; SMUTNICKI, 2005).

Figura 4.2 – Representação da função Griewangk.



Fonte: Adaptado de Molga e Smutnicki (2005).

Esta função é semelhante à função de Rastrigin, com muitos mínimos locais generalizados regularmente distribuídos (MOLGA; SMUTNICKI, 2005). Sua definição matemática é:

$$f(x) = 1 + \sum_{i=1}^{N} \frac{X_i^2}{4000} - \prod_{i=1}^{N} cos\left(\frac{X_i}{\sqrt{i}}\right). \quad (4.1)$$

4.1.2 Rastrigin

A função Rastrigin (Ras) é baseada na função de DeJong com a adição da modulação do cosseno para produzir mínimos locais frequentes (MOLGA; SMUTNICKI, 2005). Por isso, a função Ras consiste em vários mínimos locais (ILHAN, 2016), distribuídos regularmente (MOLGA; SMUTNICKI, 2005). A representação da função em espaço bidimensional é ilustrada na Figura 4.1b e sua definição matemática é:

$$f(x) = 10N + \sum_{i=1}^{N} [X_i^2 - 10\,cos(2\pi X_i)]. \quad (4.2)$$

4.1.3 Rosenbrock

O vale da Rosenbrock (Ros) é um problema de otimização clássico, também conhecido como “função banana” ou a “segunda função de DeJong”. O mínimo global está dentro de um vale plano longo, estreito e parabólico (MOLGA; SMUTNICKI, 2005; THEODOSSIOU; KOUGIAS, 2012). Encontrar o vale é trivial, porém a convergência para o ótimo global é difícil (MOLGA; SMUTNICKI, 2005), ainda mais quando a função se expande em 30 dimensões e consiste em 30 variáveis (THEODOSSIOU; KOUGIAS, 2012) (Figura 4.1c). Esta função, que tem sido frequentemente usada para testar o desempenho de técnicas de otimização, é definida por:

$$f(x) = \sum_{i=1}^{N-1} \left[100\left(X_i^2 - X_{i-1}\right)^2 + (1 - X_i)^2\right]. \quad (4.3)$$

4.1.4 Schwefel

A função de Schwefel (Sch) é uma função que tem o segundo melhor mínimo longe do mínimo global (ILHAN, 2016). É uma função enganosa na medida em que o mínimo global é geometricamente distante, no espaço de projeto, do próximo melhor mínimo local (Figura 4.1d). Portanto, os algoritmos de busca são potencialmente propensos à convergência na direção errada (MOLGA; SMUTNICKI, 2005). Sua definição é dada por:

$$f(x) = 418{,}989N - \sum_{i=1}^{N} X_i\, sin\left(\sqrt{|X_i|}\right). \quad (4.4)$$

4.1.5 Ackley

Ackley (Ack) é uma função de teste multimodal amplamente utilizada (ILHAN, 2016; MOLGA; SMUTNICKI, 2005). Em sua forma bidimensional (Figura 4.1e) a função Ack é caracterizada por uma região externa quase plana e um grande vale no centro. Assim, a função representa um risco para alguns algoritmos de otimização, que podem ficar presos em um de seus muitos mínimos locais (SURJANOVIC; BINGHAM, 2013). A definição da função é dada por:

$$f(x) = 20 + exp\ -20\,exp\left(-0{,}2\sqrt{\frac{1}{N}\sum_{i=1}^{N} X_i^2}\right) - ex\,p\left(\frac{1}{N}\sum_{i=1}^{N} cos(2\pi X_i)\right). \qquad (4.5)$$

4.1.6 Beale

A função Beale (Bea) é multimodal, com picos agudos nos cantos do domínio de entrada (Figura 4.1f), e sua definição é dada por:

$$f(x) = (1{,}5 - x_1 + x_1x_2)^2 + (2{,}25 - x_1 + x_1x_2^2)^2 + (2{,}625 - x_1 + x_1x_2^3)^2. \qquad (4.6)$$

4.2 Funções CEC 2017

A avaliação de desempenho dos algoritmos utilizou um conjunto com 30 funções da competição CEC 2017 denominada Real-Parameter Single Objective Optimization. As características das funções são divididas em quatro tipos: Unimodal, Multi-modal, Híbrido, e Funções de composição. As funções unimodais são inseparáveis e rotativas e possuem um único ótimo global sem qualquer ótimo local. As funções multimodais são rotativas e/ou deslocadas e também são separáveis ou não separáveis. As funções híbridas são desenvolvidas de tal forma que as variáveis são divididas aleatoriamente em alguns subcomponentes. Por fim, as funções de composição consistem na soma de duas ou mais funções básicas.

O conjunto de 30 funções é apresentado resumidamente na Tabela 4.2 e o intervalo de busca das variáveis de projeto é limitado a x i,j ∈ [-100, 100] para todas as funções. Os problemas B1 a B3 são unimodais, B4 a B10 são multimodais, B11 a B20 são híbridos e B21 a B30 são funções de composição.

Tabela 4.2. Resumo das funções teste CEC 2017.

No.	Funções	f(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

(continua)

Tabela 4.2. Conclusão.

No.	Funções	f(x)*
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

*Execpionalmente no CEC2017, o problema B2 foi excluído devido ao seu comportamento instável e por apresentar variações no desempenho para um mesmo algoritmo (AWAD et al., 2017).

Fonte: Adaptado de Awad et al. (2017).

Todas as funções utilizadas possuem como ótimo global o valor 0. Porém, para fins de apresentação e organização, o CEC realizou o deslocamento do ótimo global de cada função por 100 vezes o número da função. Ou seja, a função 1 possui ótimo global valendo 100, a função 2 possui o ótimo global valendo 200. Para fins de comparação entre os algoritmos neste trabalho, o ótimo global de todas as funções utilizado foi o 0. Essa opção foi a mesma utilizada no trabalho de (BARROCA, 2019) e permite observar o quão próximo de 0 o valor médio da função objetivo atinge ao final das buscas.

Os algoritmos a serem testados utilizando esse conjunto de funções foram executados de forma independente para todas as funções por 51 vezes e o critério de parada foi definido como 10000 × D, onde D é a dimensão do problema. Ao final das 51 execuções, foi calculada a média do valor da função objetivo atingido. A função de número 2 não foi executada devido ao seu comportamento instável e por apresentar variações em seu desempenho.

As dimensões D testadas foram 10 e 30 e representam a quantidade de variáveis de projeto. Portanto, o critério de parada definido foi de 100.000 NFEs e 300.000 NFEs, respectivamente. Tendo em vista que as versões binárias codificam as variáveis de projeto em uma população de bits, foi definido que estas versões utilizassem 16 bits para representar cada variável. Além disso, o espaço de busca definido para todas as funções foi o mesmo utilizado na CEC, que foi o intervalo entre -100 e +100.

4.3 Implementação numérica do problema conceitual de sistema espacial

O ótimo global foi definido como critério de parada para as simulações e 100 execuções independentes foram realizadas para cada algoritmo. Se o MDO atingiu o critério de parada, os resultados são retornados e o processo é finalizado. Caso contrário, uma nova avaliação é calculada a partir de um novo ponto selecionado pelo otimizador.

Para a execução, as versões binárias utilizaram 2, 6 e 6 bits para codificar as variáveis I, D e Q, respectivamente. Foram realizadas 100 execuções

independentes para cada algoritmo e o critério de parada foi definido como 100.000 NFEs ou então quando atingir o ótimo global. Ao fim, foi realizada a média aritmética dos NFEs obtidos em cada execução. Por exemplo, se são feitas somente 2 execuções e em uma delas o algoritmo atingiu o ótimo global em 1000 NFEs e em outra ele não atingiu o ótimo global (atingiu 100.000 NFEs), a média será de 55.000 NFEs.

4.4 Ambiente de desenvolvimento

Todas as implementações realizadas neste trabalho utilizaram uma máquina com as especificações apresentadas na Tabela 4.2 e foram desenvolvidas com a linguagem C#. Para isso, foi utilizado o .NET 5.0 que é uma plataforma de desenvolvimento atualmente open-source, criada pela Microsoft, que fornece facilidades de reutilização e reaproveitamento de código e foi desenvolvida para disponibilizar um ambiente único para desenvolvimento e execução de sistemas e aplicações. Essa ferramenta foi utilizada em conjunto com o software Visual Studio Code (VS Code) que é um editor de textos de código aberto, o que permitiu a edição dos códigos, execução e depuração.

Tabela 4.3. Especificações do hardware onde os experimentos foram executados.

Hardware	**Descrição**
Processador	Intel© Core™ i5-7200U CPU @ 2.50GHz x 2
Memória RAM	8 GB
Sistema Operacional	Linux Mint 20.2 Cinnamon 64-bits
Kernel Linux	5.13.0-25-generic x86_64 bits
Drive	SSD 240 GB + HD 1TB

Fonte: Produção do autor.

Além da implementação dos algoritmos com codificação binária e real, foi desenvolvido um software que permite executar por N vezes os algoritmos

escolhidos e calcular a média dos resultados finais, além de realizar o ajuste de parâmetros para os algoritmos escolhidos, o que totalizou 6621 linhas de código. Por fim, é importante observar que os tempos de execução dos algoritmos estão diretamente relacionados com a lógica do código, ou seja, um futuro melhoramento de desempenho do código permitiria melhorar o tempo de processamento sem precisar migrar para uma outra linguagem de programação, ao por exemplo utilizar técnicas de processamento paralelo.

As populações binárias e reais estudadas neste trabalho foram implementadas de maneiras diferentes no código. Enquanto que a população binária foi representada como um vetor booleano, a população real foi representada através de um vetor de valores ponto flutuante. Para isso, foi utilizado o tipo de dado Double do .NET, que representa valores com uma precisão de até 16 dígitos.

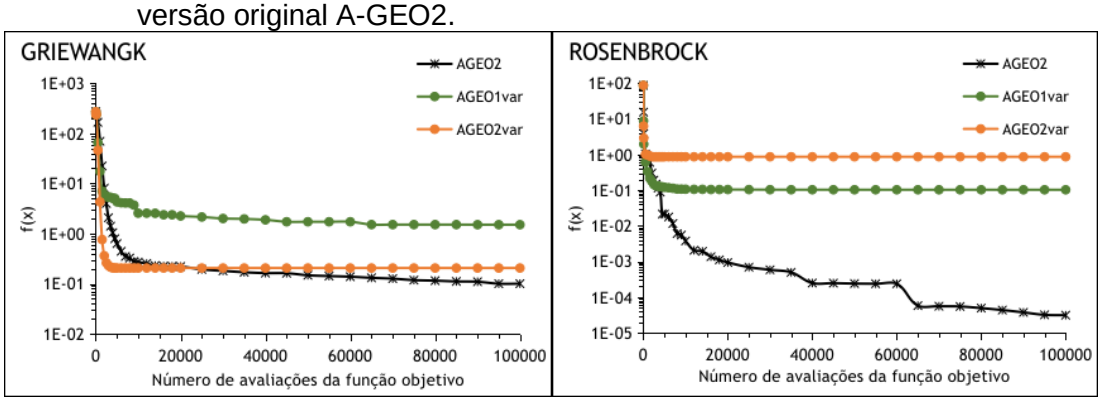
5 RESULTADOS EXPERIMENTAIS UTILIZANDO FUNÇÕES TESTE

Este Capítulo destina-se a apresentar os resultados experimentais obtidos com as implementações numéricas e compreende as avaliações de desempenhos de todos os algoritmos estudados.

5.1 Avaliação do desempenho do A-GEOvar

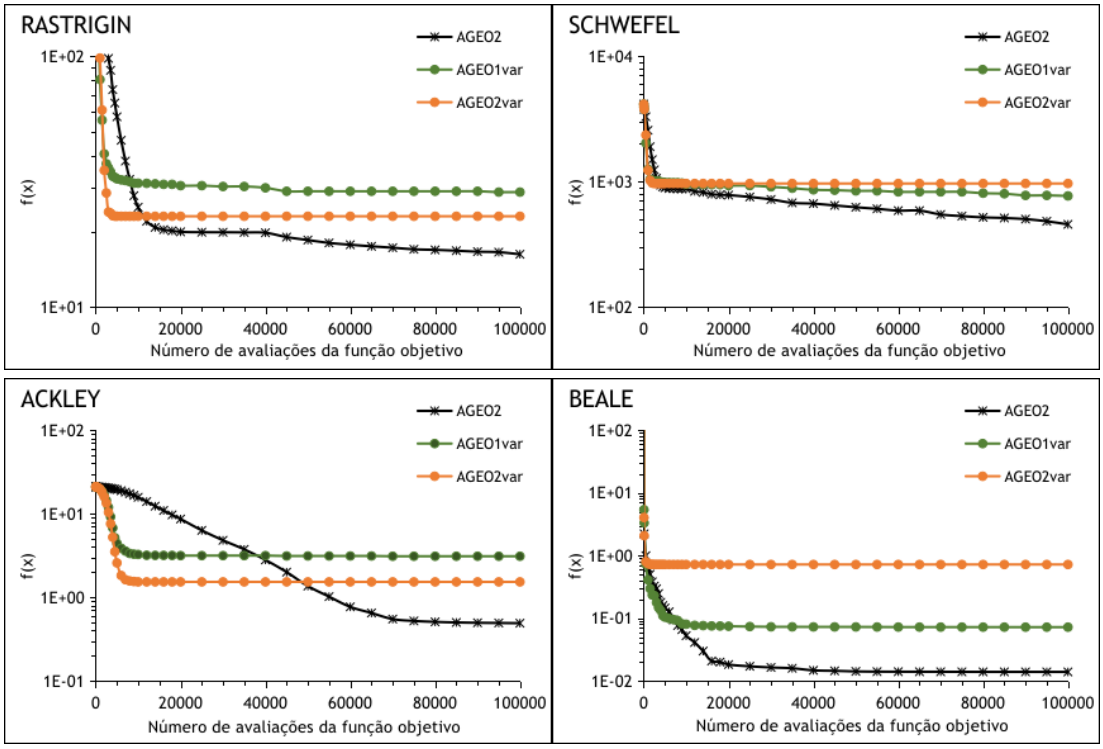
Os resultados apresentados por Barroca (2019) mostraram que a versão A-GEO2 foi a melhor versão do GEO adaptativo, sendo essa superior à versão A-GEO1 para as funções testadas em seu trabalho. Tendo em vista o melhor desempenho do GEOvar com relação ao GEO, esperava-se um melhor desempenho do A-GEOvar em comparação ao A-GEO. A implementação da versão adaptativa do algoritmo GEOvar gerou as versões A-GEO1var e A-GEO2var. Essas versões foram comparadas com a versão A-GEO2 e os gráficos de desempenho são apresentados na Figura 5.1. Cada gráfico representa o valor médio da função objetivo nas 50 execuções em função do NFE.

Figura 5.1. Comparativo de desempenho das versões A-GEO1var e A-GEO2var com a versão original A-GEO2.



(continua)

Figura 5.1. Conclusão.



Fonte: Produção do autor.

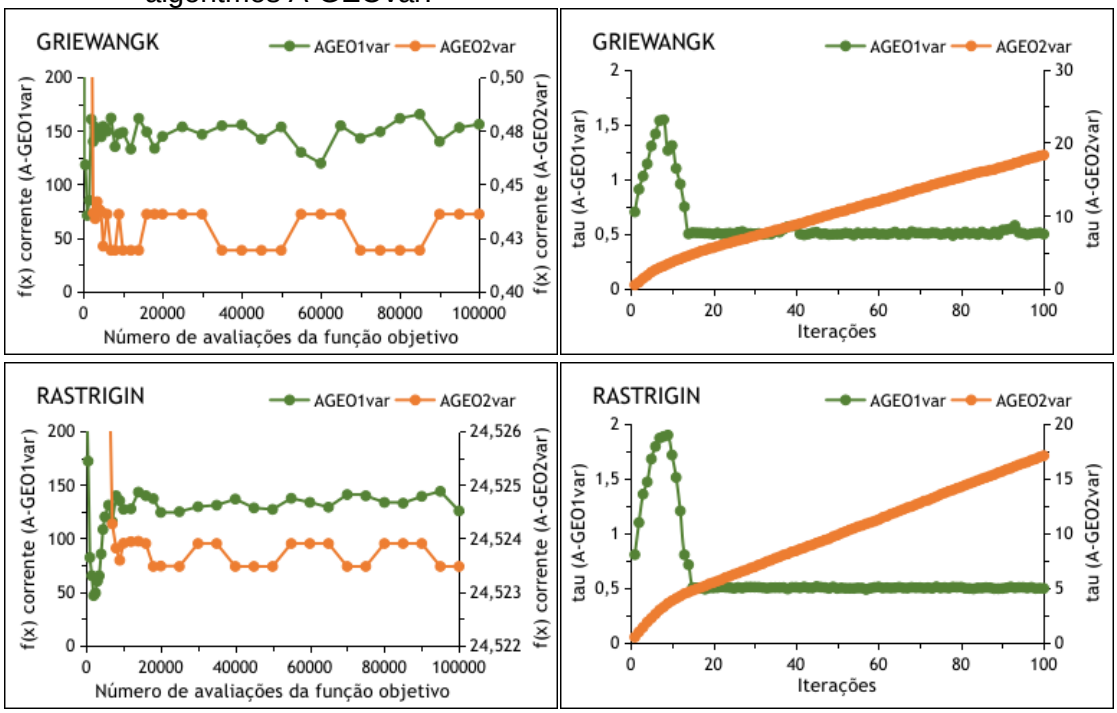
Inicialmente, é possível observar um melhor desempenho do A-GEO2 para todas as funções teste. Isso vai ao contrário da hipótese em que uma versão adaptativa do GEOvar seria superior ao A-GEO. Tendo em vista que no GEOvar a solução é calculada somente após mutar 1 bit por variável, é possível que o algoritmo esteja estagnado e a busca esteja permutando somente entre dois pontos do espaço de projeto. Observa-se que o A-GEO2var não apresenta nenhuma melhora em todas as funções após cerca de 5000 NFEs.

O GEOvar possui como característica a mutação de 1 bit para cada variável de projeto. Portanto, uma situação onde os mesmos bits são mutados a cada iteração é muito provável de ocorrer uma vez que a busca está muito determinística. Caso uma mutação em uma variável melhore a solução em comparação com a solução de referência e uma mutação de outra variável não melhore, mas seja a melhor mutação daquela variável, ambos os bits são mutados, diferentemente do A-GEO2 onde somente 1 bit da população inteira é

mutado. Aliado a isso, sempre que ao menos uma mutação de bit melhora a solução em comparação à solução de referência, o CoI nunca será 0 e o τ nunca será resetado, tornando a busca cada vez mais determinística.

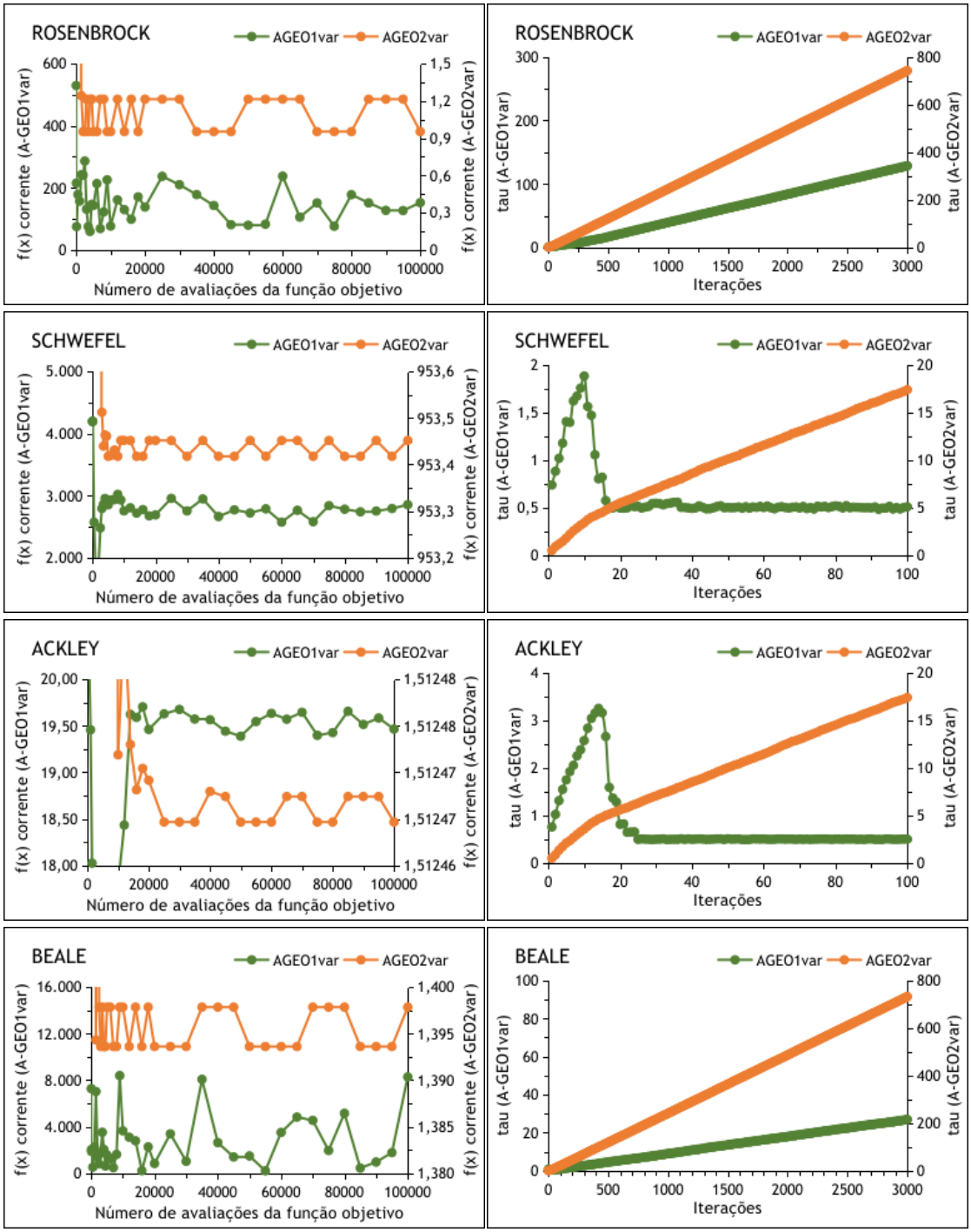
Para verificar de forma mais profunda a dinâmica dos algoritmos A-GEO1var e A-GEO2var, é necessário analisar as curvas que mostram o valor da função objetivo da população atual a cada NFE, bem como o valor de τ a cada iteração. Esses gráficos foram gerados e são apresentados na Figura 5.2. Os resultados apresentam a média de 50 execuções independentes. À esquerda, situam-se os gráficos que descrevem a média do valor da solução corrente (cada algoritmo em um eixo vertical utilizando escala linear) em função do NFE. À direita, situam-se os gráficos que descrevem o valor médio de τ (cada algoritmo em um eixo vertical utilizando escala linear) em função do número de iterações.

Figura 5.2. Comparativo da população corrente média e do valor médio de τ entre os algoritmos A-GEOvar.



(continua)

Figura 5.2. Conclusão.



Fonte: Produção do autor.

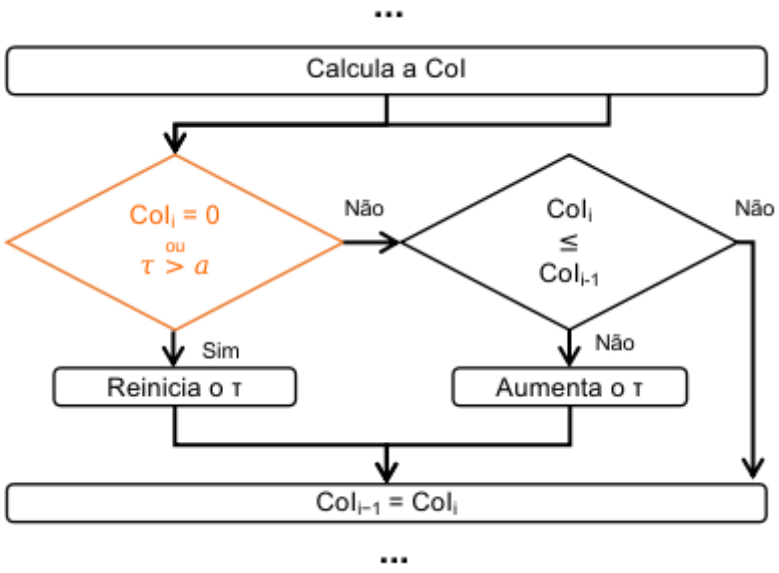
É possível observar nos gráficos à esquerda que, a partir de aproximadamente 20 mil NFEs, a população corrente do A-GEO2var alternou exatamente entre dois pontos do espaço de projeto. Os gráficos à direita mostram que o valor de

τ deste algoritmo cresceu de uma forma aproximadamente linear, tornando a busca cada vez mais determinística e não permitindo o algoritmo realizar uma maior utilização de mutação para explorar o espaço de projeto.

Ao ser comparado com o A-GEO2var, observa-se que a população corrente do A-GEO1var atingiu muitas variações no começo da busca até a população se estabilizar, mas estes valores não estagnaram o algoritmo em apenas 2 pontos do espaço de projeto como na outra versão. Os gráficos de τ em função do número de iterações permitem investigar que seu valor médio para as funções Rosenbrock e Beale cresceram, mas não com a mesma intensidade do A-GEO2var. Isso ocorreu devido ao τ crescer aproximadamente linearmente até o fim da busca em algumas execuções e do τ ser muito resetado em outras execuções, fazendo com que na média o τ seja aproximadamente linear para essas duas funções. Para as outras 4 funções, o valor de τ cresceu no começo das 50 buscas e depois foi sempre resetado (representando uma linha aproximadamente horizontal nos gráficos). Isso mostra que somente nas funções Ros e Bea o A-GEO1var teve algumas execuções onde o τ cresceu sem ser resetado, fazendo com que o algoritmo não conseguisse explorar o espaço de projeto.

O mecanismo adaptativo determina que o τ somente seja resetado quando o valor de CoI é 0, e para isso ocorrer é preciso que nenhuma das mutações provenha uma melhor solução do que a solução corrente no A-GEO2var. É possível que, dentre todas as mutações para cada variável de projeto, ao menos uma esteja provendo uma solução melhor que a de referência, fazendo que CoI seja maior que 0 e o τ não seja resetado. Portanto, no momento em que a mutação de um bit para cada variável de projeto é aceita, uma variável pode estar melhorando a solução e outra piorando. Tendo em mente essa estagnação da versão A-GEO2var, uma maneira de melhorar seu desempenho seria alterar o mecanismo adaptativo para forçar resetar o valor de τ sempre que CoI for 0 ou então quando o valor de τ ultrapassar um limiar α, passando a tornar a busca mais estocástica novamente e permitindo a exploração. Com isso, o mecanismo teria apenas uma modificação na primeira condição, conforme apresentado na Figura 5.3.

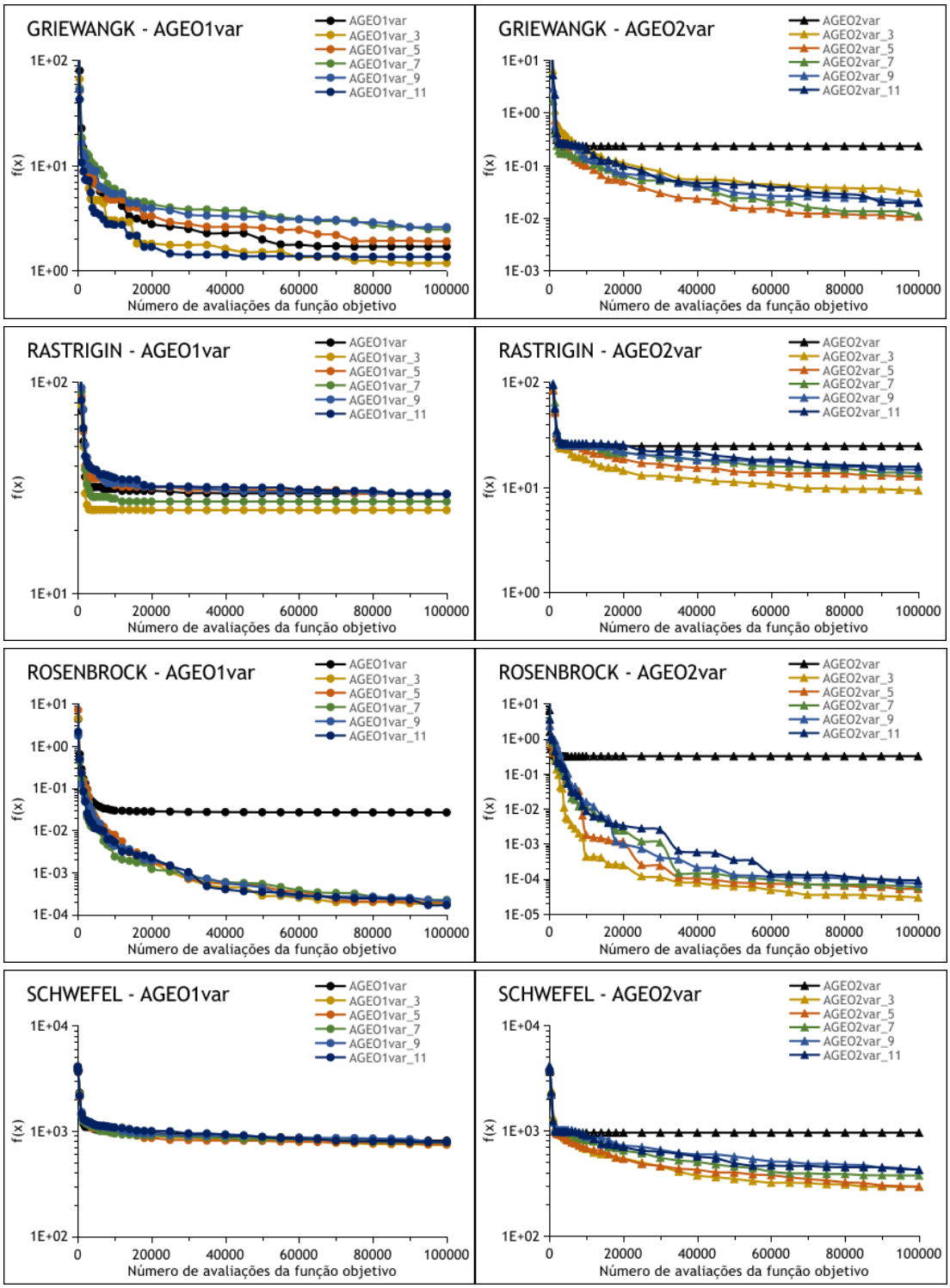
Figura 5.3. Modificação no mecanismo adaptativo para o A-GEOvar.



Fonte: Produção do autor.

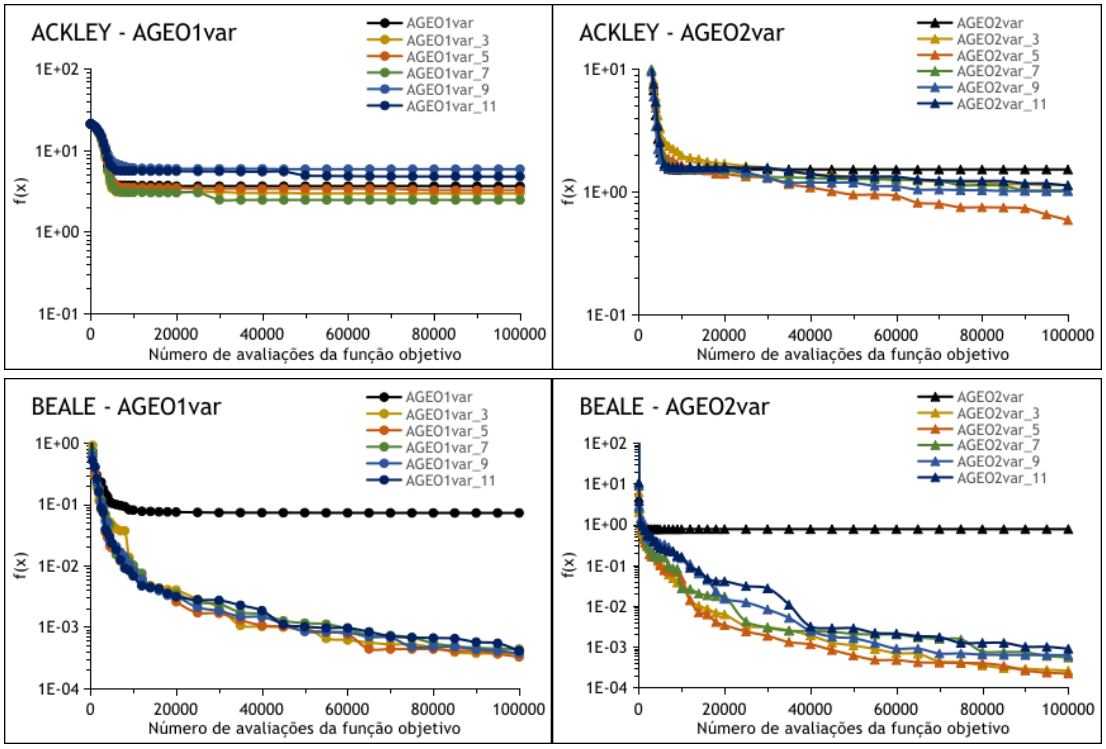
Diferentes valores foram testados para o parâmetro α, visando encontrar um valor para resetar o parâmetro τ que atingisse o melhor desempenho dentre os parâmetros testados. Esse valor não pode ser muito baixo para que o τ não resete continuamente, não permitindo a intensificação da busca, e também não pode ser muito alto para que a busca não se torne muito determinística antes de resetar. Essa modificação foi realizada no mecanismo adaptativo para as versões A-GEO1var e A-GEO2var e foram testados os valores 3, 5, 7, 9 e 11 para o parâmetro α. Portanto, esses 5 valores testados para as 2 versões do A-GEOvar geraram 10 variações e o comparativo de desempenho entre essas versões é apresentado na Figura 5.4. À esquerda, são apresentados os desempenhos das variações do A-GEO1var e à direita estão dispostos os desempenhos das variações do A-GEO2var.

Figura 5.4. Desempenho do A-GEOvar com o mecanismo modificado.



(continua)

Figura 5.4. Conclusão.



Fonte: Produção do autor.

É evidente o melhor desempenho que a modificação no mecanismo adaptativo resultou para a versão A-GEO2var. Para essa versão, em todas as funções, forçar resetar o τ foi mais vantajoso do que utilizar o mecanismo adaptativo original, permitindo o algoritmo aprimorar a busca e melhorar seu desempenho. Além disso, nota-se um bom desempenho da versão A-GEO2var_5, sendo essa versão a que obteve o melhor ou o segundo melhor desempenho para todas as funções avaliadas.

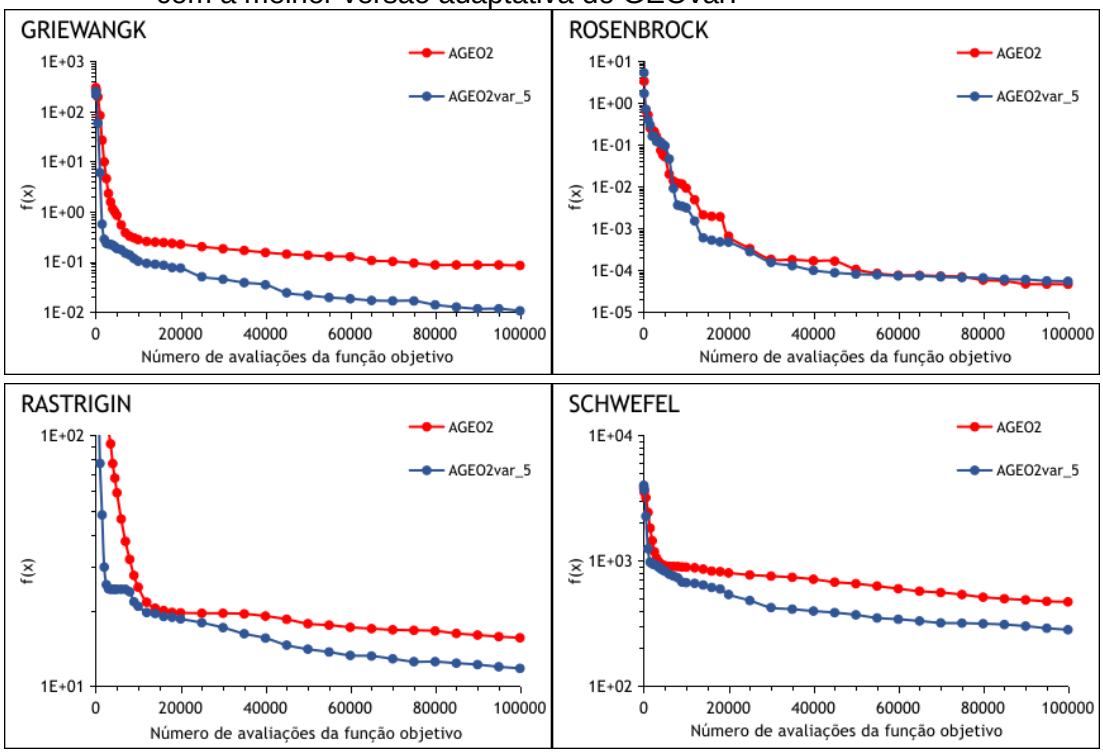
Um ganho de desempenho com a modificação do mecanismo não é observado no A-GEO1var para todas as funções teste, exceto as funções Ros e Bea. A avaliação do valor de τ médio em função do número de iterações realizado na Figura 5.1 permitiu observar que nessas duas funções o A-GEO1var teve algumas execuções onde a busca ficou estagnada, o que é possível observar novamente nos gráficos da Figura 5.4. Ao comparar as versões modificadas, nota-se um melhor desempenho da versão A-GEO1var_3, sendo essa a versão

que atingiu o melhor ou o segundo melhor desempenho dentre as versões, mostrando que forçar resetar o τ quando ele é maior que 3 é mais vantajoso.

A definição de duas novas versões para o A-GEOvar, denominadas A-GEO1var_3 e A-GEO2var_5, melhorou significativamente o desempenho da versão adaptativa do GEOvar, tanto em convergência quanto em valor da função objetivo. A modificação no mecanismo permitiu atingir uma melhora de até 10^4 no valor final da função objetivo, como observado nas funções Rosenbrock e Beale. Porém, ao comparar as versões A-GEO1var_3 com A-GEO2var_5, verifica-se um desempenho superior da segunda versão, que utiliza α = 5 e a solução de referência utilizada no cálculo do CoI é a solução corrente.

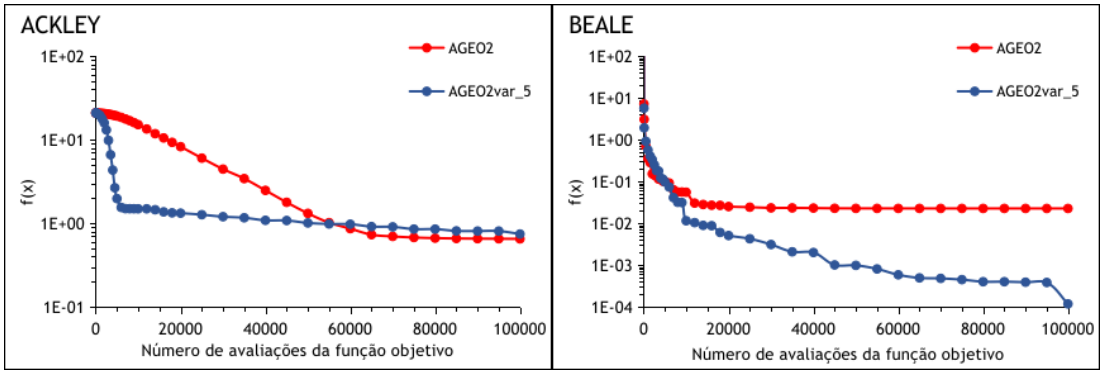
O melhor desempenho atingido pelo A-GEO2var_5 comparado ao A-GEO2var permite uma nova avaliação de desempenho deste comparado ao A-GEO2. Esse comparativo de desempenho é apresentado na Figura 5.5.

Figura 5.5. Comparativo de desempenho entre a melhor versão adaptativa do GEO com a melhor versão adaptativa do GEOvar.



(continua)

Figura 5.5. Conclusão.

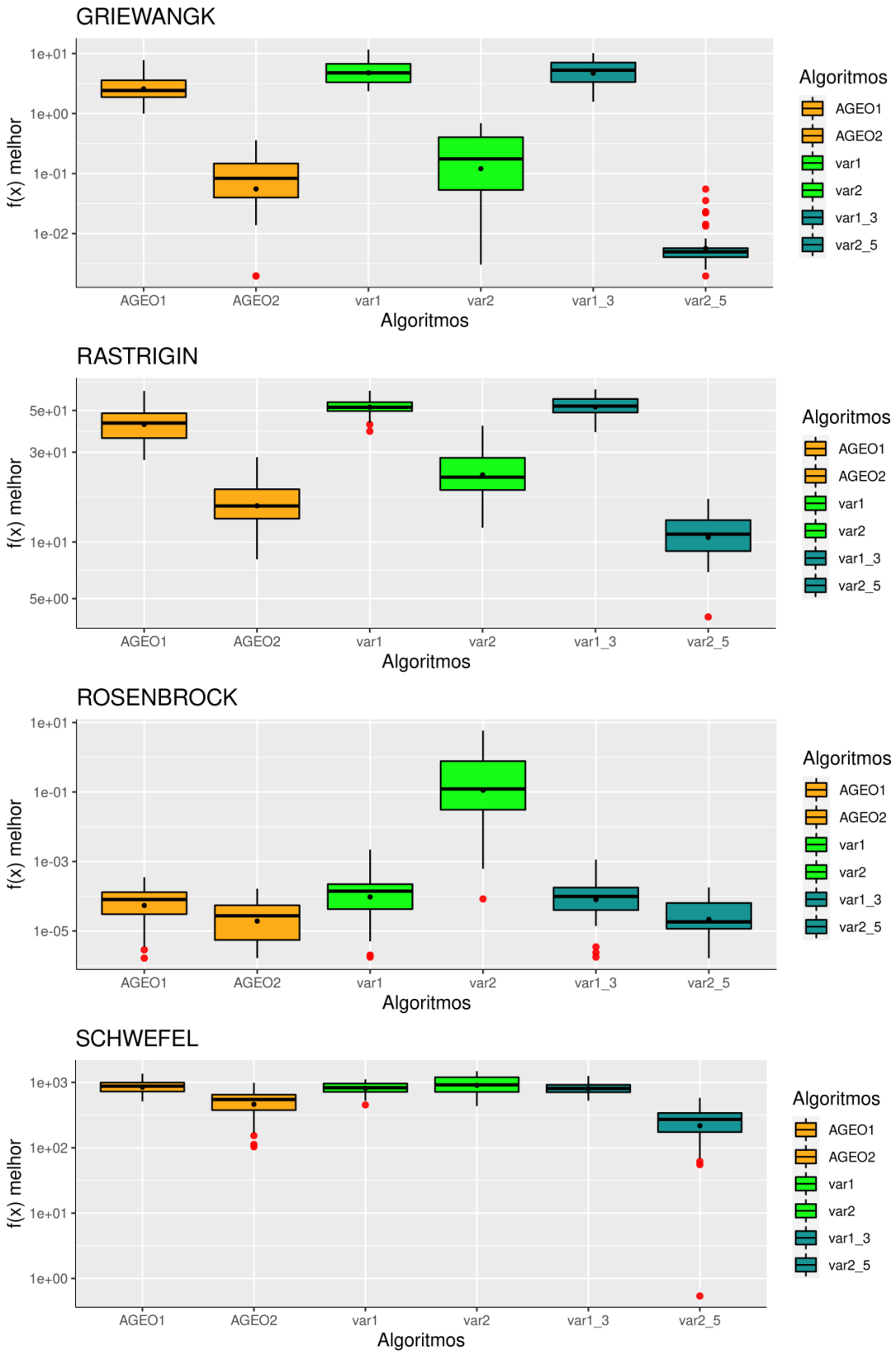


Fonte: Produção do autor.

Com a adaptação para forçar a reinicialização do τ, o algoritmo A-GEO2var_5 teve um desempenho igual ou superior ao A-GEO2 para todas as funções teste. A única função na qual foi atingido um desempenho semelhante foi a Ackley, onde o A-GEO2 superou por pouco a versão adaptativa do GEOvar na média do melhor valor da função objetivo, mas em termos de convergência a implementação A-GEO2var_5 foi superior.

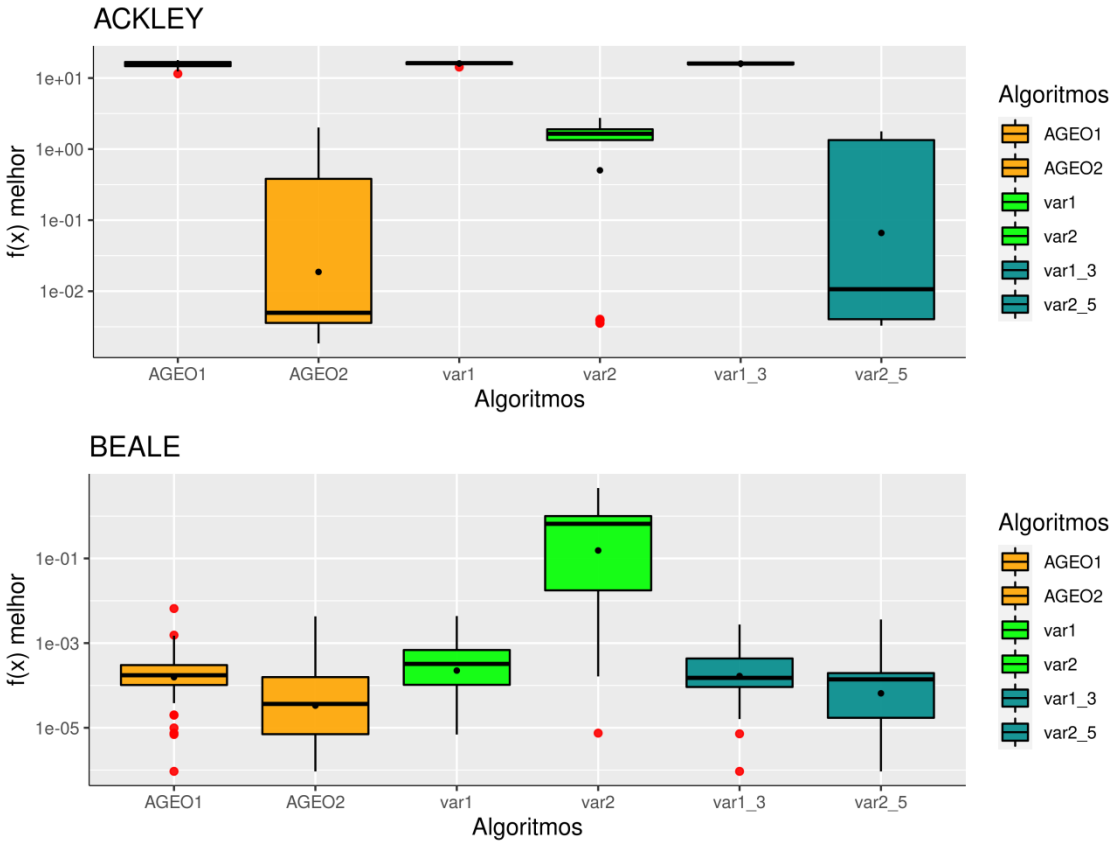
Uma análise estatística dos valores atingidos ao final das execuções foi realizada e foi gerado um boxplot para cada função comparando os diferentes algoritmos. Esta análise é apresentada na Figura 5.6 e foram comparados os algoritmos A-GEO1, A-GEO2, A-GEO1var, A-GEO2var, A-GEO1var_3 e A-GEO2var_5. Os boxplots para cada função estão na escala logarítmica.

Figura 5.6. Boxplots dos melhores valores obtidos ao final das 50 execuções para o A-GEOvar.



(continua)

Figura 5.6. Conclusão.



Todos os gráficos estão em escala logarítmica e os nomes AGEO1, AGEO2, var1, var2, var1_3 e var2_5 representam, respectivamente, os algoritmos A-GEO1, A-GEO2, A-GEO1var, A-GEO2var, A-GEO1var_3 e A-GEO2var_5.

Fonte: Produção do autor.

Os resultados da Figura 5.6 corroboram com os resultados apresentados na Figura 5.1, uma vez que é possível observar que o A-GEO2var atingiu um desempenho inferior ao A-GEO2 para todas as funções objetivo. Além disso, a mudança no mecanismo que gerou o A-GEO2var_5 atingiu melhores resultados que o A-GEO2var em todas as funções. Para estes dois algoritmos, observa-se na função Ackley que, embora a dispersão do boxplot tenha sido grande no A-GEO2var_5, tanto a média aritmética quanto a mediana dos resultados foi melhor que a do A-GEOvar.

Essa avaliação conclui que uma versão adaptativa do GEOvar é mais vantajosa do que a versão adaptativa do GEO, conforme também concluído no trabalho de (DA LUZ et al., 2021). Embora a simples implementação do

mecanismo adaptativo no GEOvar não atingiu um melhor desempenho, uma modificação em uma das condições do mecanismo permitiu a nova implementação alcançar um bom desempenho.

5.2 Avaliação do desempenho do GEOreal1

As maneiras Porcentagem e Aditiva de perturbar as variáveis foram comparadas com a mutação Multiplicativa e 3 versões do GEOreal1 foram testadas, denominadas GEOreal1_M (utilizando mutação Multiplicativa), GEOreal1_P (utilizando mutação Porcentagem) e GEOreal1_A (utilizando mutação Aditiva). Todos os algoritmos possuem como parâmetro livre o τ que determina a estocasticidade e determinísmo da busca e um outro parâmetro que age diretamente na perturbação, totalizando 2 parâmetros livres. Para as mutações Multiplicativa e Aditiva este parâmetro é σ (desvio-padrão) e para a mutação Porcentagem o parâmetro que age na mutação é o ρ (porcentagem) que representa a porcentagem do intervalo de variaçao das variáveis que será utilizado como desvio padrão na distribuição de probabilidade.

Levando em conta os 2 parâmetros livres, foi realizada a etapa de ajuste de parâmetros a fim de variar seus valores e encontrar o conjunto de valores que resulta no melhor desempenho. Para isso, foram testados valores de τ entre 0,5 a 6,0 com um intervalo a cada 0,5 e valores de σ entre 0,2 e 3,0 com um intervalo a cada 0,2. Com relação aos valores de ρ, foram testados valores entre 1% e 100,0% com um intervalo a cada 1% e foi observado que os melhores desempenhos foram obtidos com porcentagens menores ou iguais a 10%. Com isso, para um ajuste fino, foram testados valores entre 0,5% e 10,0% com um intervalo a cada 0,5. Todas as combinações possíveis entre os parâmetros foram realizadas a fim de encontrar o conjunto de parâmetros que resultasse no melhor desempenho. Os parâmetros são apresentados Tabela 5.1.

Tabela 5.1. Ajuste de parâmetros dos algoritmos GEOreal1_M, GEOreal1_P e GEOreal1_A.

Algoritmo	Parâmetro livre	Gri	Ras	Ack	Ros	Sch	Bea
GEOreal1_M	τ: Estocasticidade	1,5	1,5	1,0	5,5	5,5	6,0
	σ: Desvio padrão	1,2	1,0	0,8	2,4	1,8	2,0
GEOreal1_P	τ: Estocasticidade	5,5	5,5	6,0	6,0	3,5	4,0
	ρ: Porcentagem	0,5	3,0	1,0	0,5	9,0	2,5
GEOreal1_A	τ: Estocasticidade	1,5	5,0	6,0	5,5	1,5	2,5
	σ: Desvio padrão	2,0	0,4	0,4	0,2	1,0	0,2

Fonte: Produção do autor.

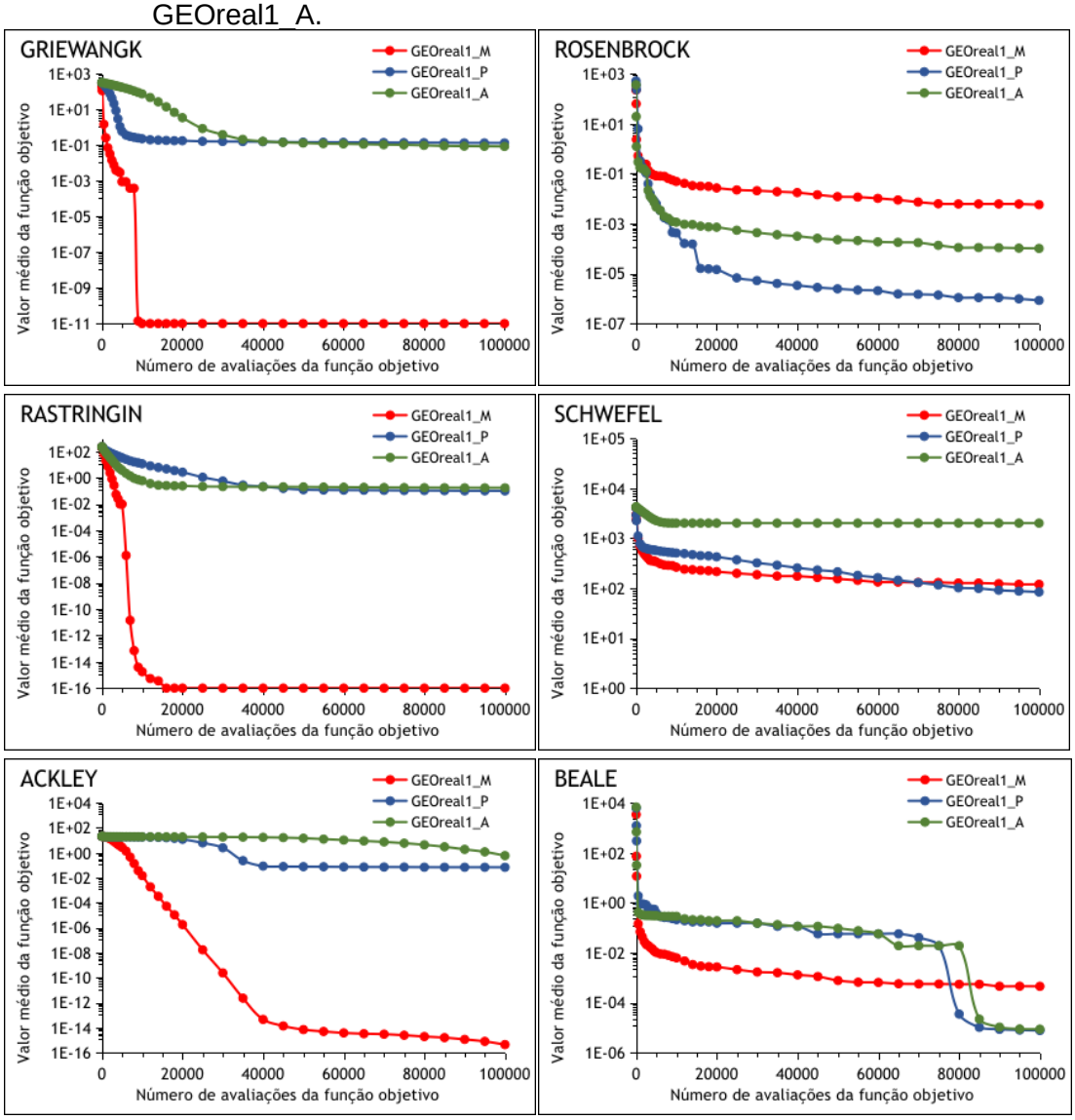
Observa-se que nas funções Grewangk, Rastrigin e Ackley onde o valor das variáveis no ótimo global vale 0 (x* = 0), o desvio padrão do GEOreal1_M ficou perto de 1,0, enquanto que para as outras 3 funções esse valor ficou perto de 2,0. Além disso, o valor de τ ajustado para as funções onde x* = 0 vale aproximadamente 1,5 enquanto que para as outras funções vale aproximadamente 5,5. Isso significa que ao utilizar o GEOreal1_M para um problema desconhecido, a implementação pode ir muito bem ou muito mal, dependendo se a função possui x* = 0. Portanto, é preferível utilizar outro algoritmo que tenha um caráter mais genérico e seja mais estável para diversos tipos de função.

Os valores de ρ para a mutação Porcentagem variaram entre 0,5% e 9,0%, o que significa que a utilização de porcentagens menores que 10% é mais benéfica do que utilizar grandes porcentagens. O maior valor foi 9,0% para a função Shwefel, provavelmente pelo fato dessa função ter o mínimo global geometricamente distante do próximo melhor mínimo local, o que leva os algoritmos a necessitarem de grandes saltos no espaço de projeto para não convergirem na direção errada.

Esses parâmetros ajustados foram configurados nos algoritmos e foram gerados gráficos de desempenho comparando os 3 algoritmos, que são apresentados na Figura 5.7. Os gráficos apresentam a média do melhor valor

da função objetivo para as 50 execuções em função do número de avaliações da função objetivo.

Figura 5.7. Comparativo de desempenho das versões GEOreal1_M, GEOreal1_P e GEOreal1_A.



Diferentes cores representam diferentes formas de perturbar as variáveis de projeto.

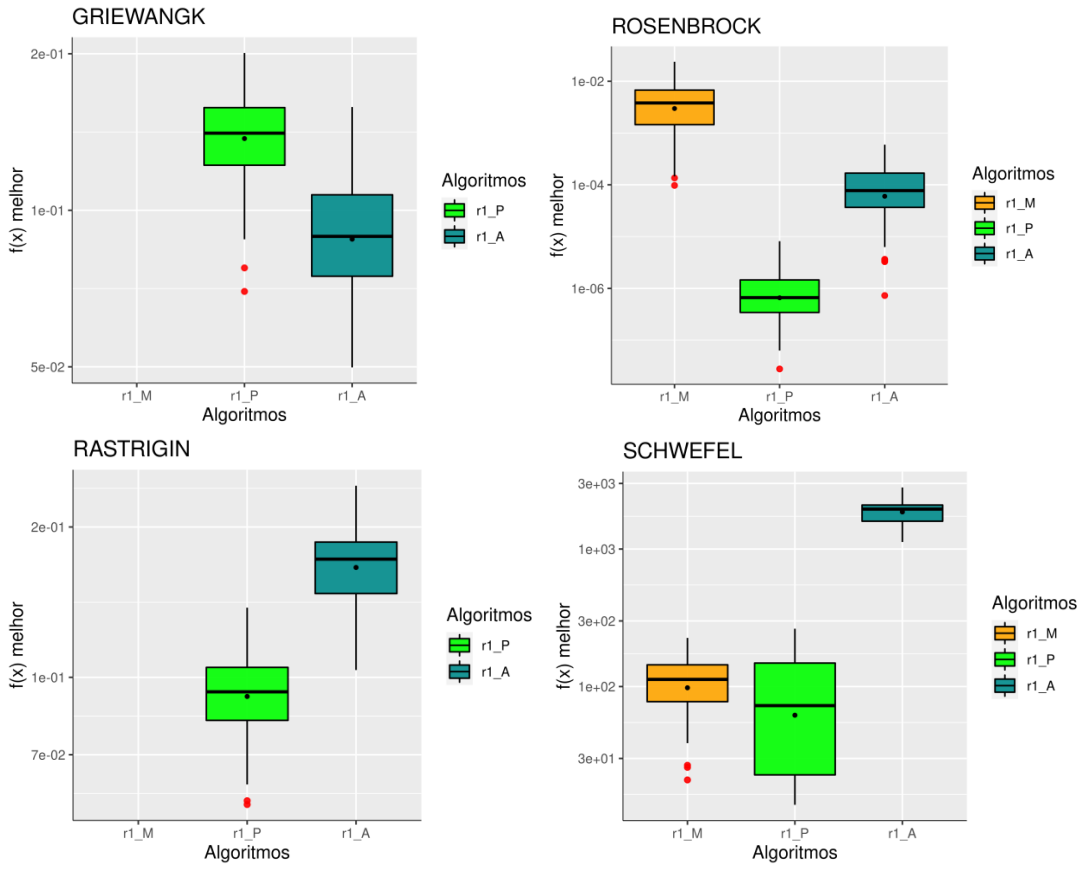
Fonte: Produção do autor.

O primeiro ponto importante de se observar é que a mutação Multiplicativa foi a melhor para as funções Gri, Ras e Ack (3 gráficos à esquerda na Figura 5.7) onde x* = 0. É importante destacar que para esse tipo de função, essa forma de perturbar as variáveis obteve desempenhos muito bons, chegando a atingir o ótimo global nas funções Gri e Ras e uma diferença de cerca de 1016 na função Ras. Assim como foi observado no trabalho de (DA LUZ et al., 2021), a

utilização da mutação Multiplicativa nas variáveis de projeto possui mais vantagem para as funções onde x* = 0, provavelmente pelo fato de que quanto menor o valor da variável, menor é a perturbação. Com isso, quanto mais próximo de 0 está o valor da variável, menor vai sendo a mutação e a capacidade de aprimoramento (exploitation) da busca é aumentada. Quando o valor das variáveis no ótimo global é muito alto, a busca continua tendo altas mutações nas variáveis mesmo perto desse valor, pois o valor da variável multiplica o valor aleatório da distribuição normal, dificultando a capacidade de exploitation na busca.

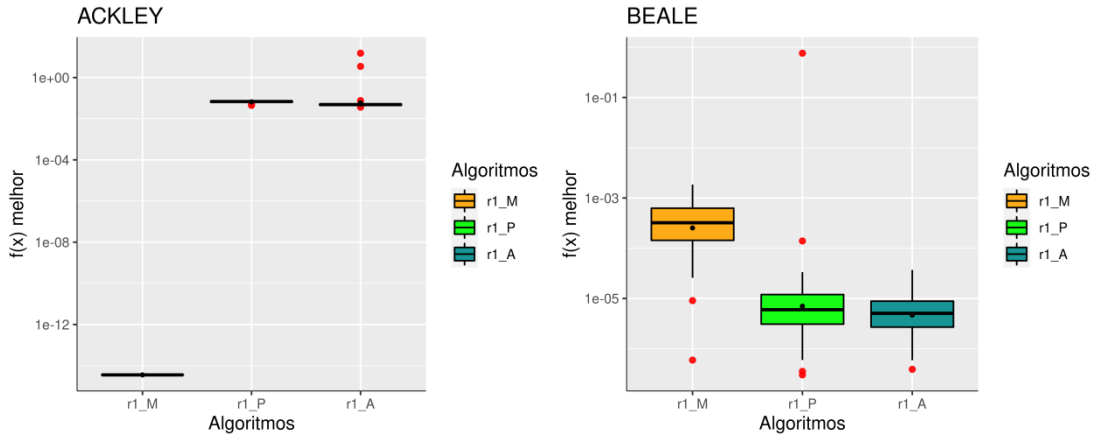
Uma análise estatística dos valores atingidos ao final das execuções do GEOreal1 foi realizada e foi gerado um boxplot para cada função comparando as três implementações (Figura 5.8). Todos os boxplots estão na escala logarítmica.

Figura 5.8. Boxplots dos melhores valores obtidos ao final das 50 execuções para o GEOreal1.



(continua)

Figura 5.8. Conclusão.



Todos os boxplots estão em escala logarítmica e os nomes r1_M, r1_P e r1_A representam, respectivamente, os algoritmos GEOreal1_M, GEOreal1_P e GEOreal1_A.

Fonte: Produção do autor.

Os boxplots confirmam os resultados apresentados na Figura 5.7, mostrando que a utilização da mutação Multiplicativa em problemas onde x* = 0 atinge o ótimo e funciona muito bem para este tipo de função. É importante pontuar que nos boxplots das funções Griewangk e Rastrigin, o GEOreal1_M atingiu 0 em todas as execuções, não sendo representado nos boxplots. Embora o valor atingido na função Ackley seja na ordem de 10-15, este valor pode ser considerado 0 (ótimo da função), uma vez que a precisão do cálculo foi de 16 casas decimais.

Nas funções onde x* ≠ 0 (3 gráficos da direita da Figura 5.7 e da Figura 5.8) nota-se um melhor desempenho da mutação P, implementada pelo algoritmo GEOreal1_P. Assim, considerando que em problemas reais o valor das variáveis em x* é desconhecido, de fato é o que se pretende achar, e que a implementação GEOreal1_P foi melhor que a GEOreal1_A no conjunto de funções teste, pode-se dizer que a implementação do GEOreal1 utilizando a mutação percentagem é a mais aproprida para utilização.

5.3 Avaliação do desempenho do GEOreal2

As seis versões implementadas para o GEOreal2 foram avaliadas através do conjunto de funções teste apresentado na Seção 3.1. Todos estes algoritmos possuem 4 parâmetros livres e os mesmos foram ajustados a fim de se obter o melhor desempenho. Para isso, foram testados valores para τ no intervalo de 0,5 a 6,0 a cada 0,5, valores para σ1 no conjunto {0.5, 1, 2, 10} e valores para P no conjunto {5 e 10}. Já os possíveis valores para s testados nas versões VO foram no conjunto {1, 2, 4} e nas versões DS foram no conjunto {2 e 10}.

Todas as combinações possíveis entre estes valores de parâmetros foram realizadas a fim de encontrar o conjunto de parâmetros que resultasse no melhor desempenho de cada algoritmo para cada função. Os resultados do ajuste de parâmetros são apresentados na Tabela 5.2.

Tabela 5.2. Ajuste de parâmetros para os algoritmos GEOreal2_M_VO, GEOreal2_P_VO, GEOreal2_A_VO, GEOreal2_M_DS, GEOreal2_P_DS e GEOreal2_A_DS.

Algoritmo	Parâmetro Livre	GRI	RAS	ACK	ROS	SCH	BEA
GEOreal2_M_VO	P: n. perturbações	5	5	5	5	10	10
	s: Variação Pert.	1	1	1	2	4	4
	σ1: Desvio padrão	2,0	2,0	1,0	2,0	2,0	1,0
	τ: Estocasticidade	2,5	5,0	4,5	4,0	5,0	4,5
GEOreal2_P_VO	P: n. perturbações	5	10	10	5	10	10
	s: Variação Pert.	4	4	4	1	4	4
	ρ1: Porcentagem	1,0	10,0	10,0	0,1	50,0	50,0
	τ: Estocasticidade	3,5	5,0	5,0	4,5	5,0	4,0
GEOreal2_A_VO	P: n. perturbações	5	10	10	5	10	10
	s: Variação Pert.	4	4	4	2	2	4
	σ1: Desvio padrão	10,0	0,5	2,0	0,5	1,0	10,0
	τ: Estocasticidade	3,5	5,0	5,0	4,0	3,5	3,5

(continua)

Tabela 5.2. Conclusão.

Algoritmo	Parâmetro Livre	GRI	RAS	ACK	ROS	SCH	BEA
GEOreal2_M_DS	P: n. perturbações	5	5	5	10	10	5
	s: Variação Pert.	2	2	2	2	10	10
	σ1: Desvio padrão	2,0	2,0	1,0	2,0	2,0	1,0
	τ: Estocasticidade	3,0	4,5	4,5	4,5	4,5	4,5
GEOreal2_P_DS	P: n. perturbações	10	10	10	5	10	5
	s: Variação Pert.	10	10	10	2	10	10
	ρ1: Porcentagem	1,0	10,0	10,0	0,1	50,0	50,0
	τ: Estocasticidade	4,0	5,0	4,5	4,5	4,5	4,5
GEOreal2_A_DS	P: n. perturbações	10	10	10	10	10	5
	s: Variação Pert.	10	10	10	2	10	10
	σ1: Desvio padrão	10,0	0,5	2,0	0,5	1,0	10,0
	τ: Estocasticidade	4,0	5,0	5,0	5,0	5,0	4,0

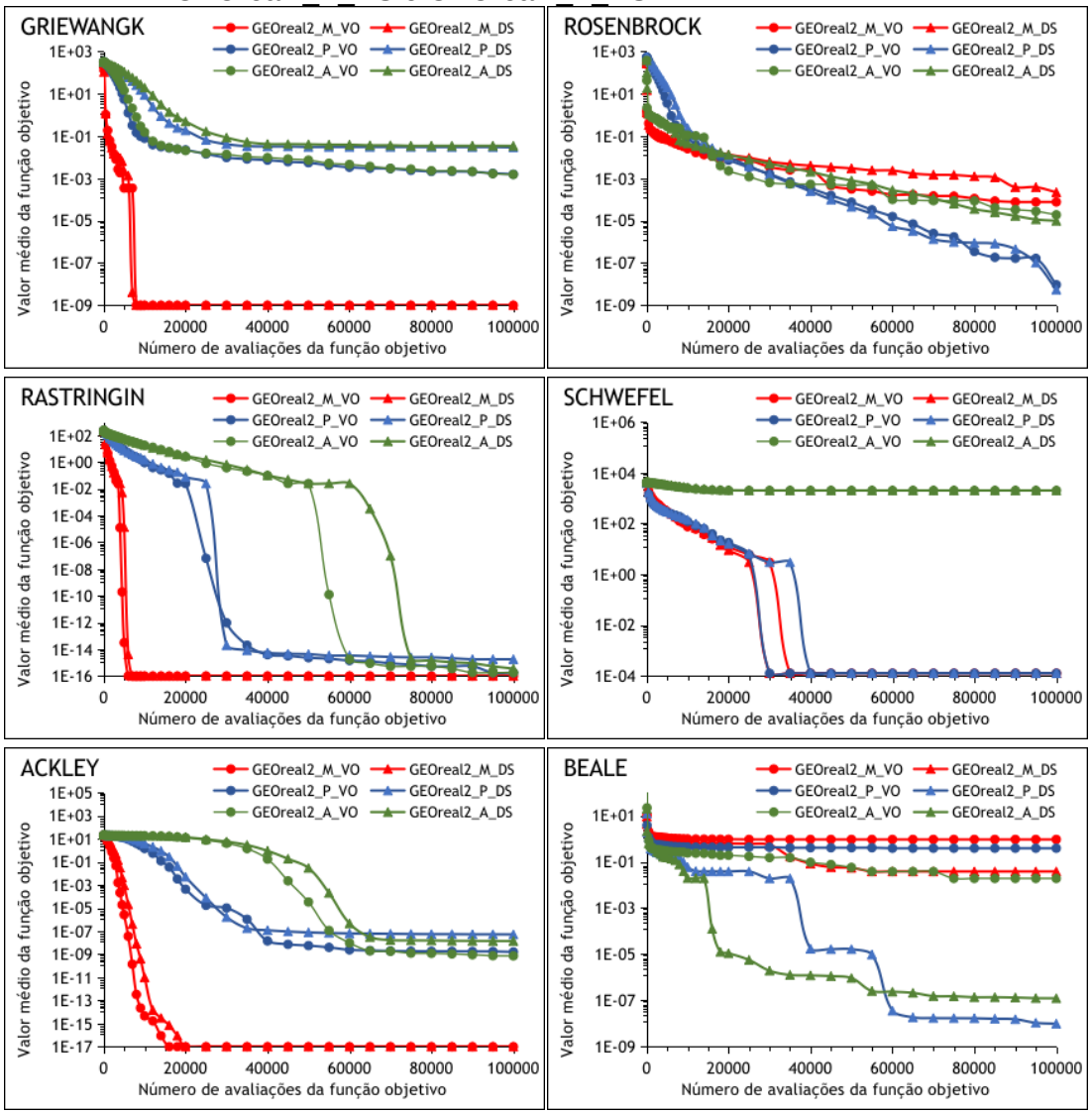
Fonte: Produção do autor.

Observa-se que o valor de P para o ajuste de parâmetros alternou entre 5 e 10. Isso indica que é interessante realizar entre 5 e 10 mutações. Menos que 5 mutações seria pouco para conseguir explorar o espaço de projeto e mais mutações que 10 não geraria muitas soluções que diferem das já geradas nas P mutações.

Nota-se uma grande diferença nos valores dos parâmetros σ1, P e s entre a mutação M e as mutações P e A, possivelmente pela maneira na qual esse tipo de mutação funciona. Além disso, é evidente a diferença que ocorre nos parâmetros que utilizam a mutação M ao comparar funções onde $x^* = 0$ e funções onde $x^* \neq 0$, principalmente nos parâmetros P e s. Isso confirma a hipótese de que essa maneira de perturbar as variáveis favorece um tipo de função e seria necessário conhecer o resultado antes de escolher aplicar o algoritmo. Portanto, espera-se um melhor desempenho da mutação M para funções onde $x^* = 0$.

Os parâmetros ajustados foram configurados nos algoritmos e um comparativo de desempenho foi executado, aplicando os 6 algoritmos para todas as 6 funções teste. Os gráficos da Figura 5.9 apresentam a média do melhor valor da função objetivo em função do NFE.

Figura 5.9. Comparativo de desempenho das versões GEOreal2_M_VO, GEOreal2_P_VO, GEOreal2_A_VO, GEOreal2_M_DS, GEOreal2_P_DS e GEOreal2_A_DS.



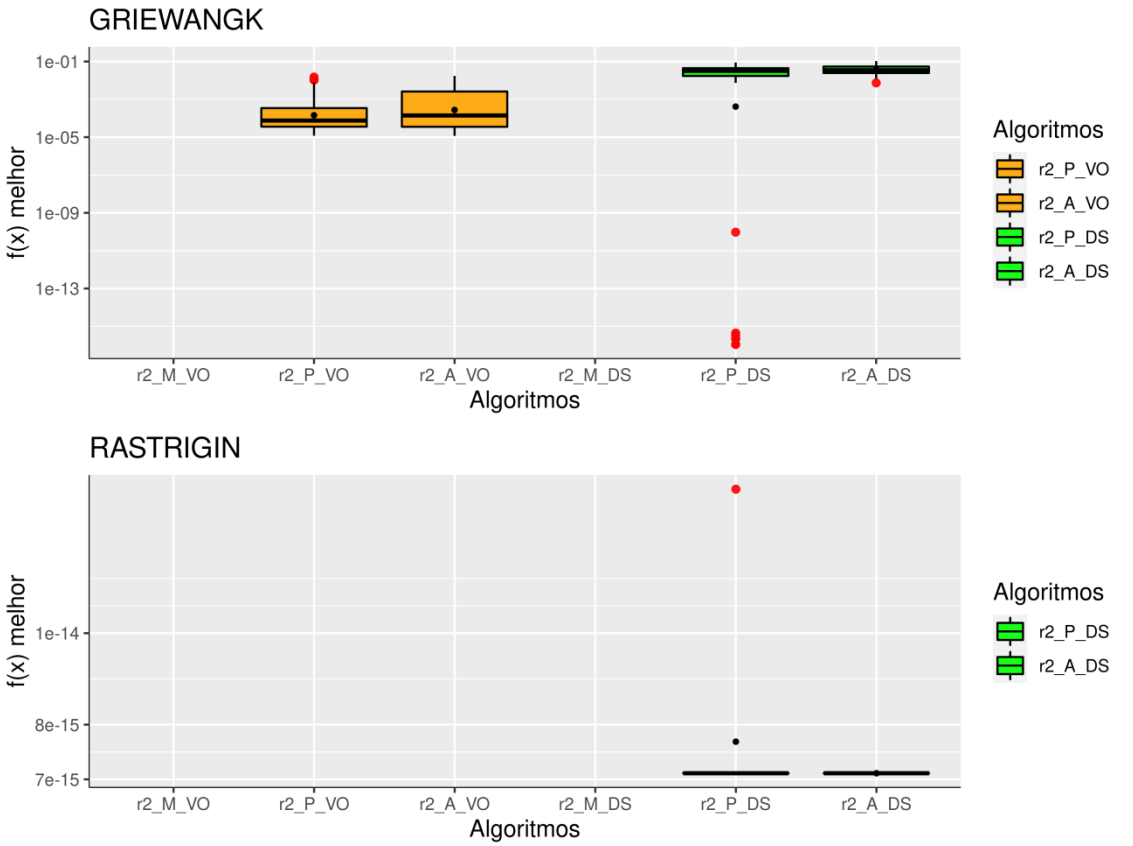
Diferentes cores representam diferentes tipos de mutação e diferentes marcadores representam diferentes formas de variar os P desvios padroes.

Fonte: Produção do autor.

Os gráficos da esquerda (funções onde x* = 0) confirmam que a mutação M atinge um desempenho melhor neste tipo de função. Além disso, percebe-se que os algoritmos que utilizam essa forma de mutação obtiveram os piores resultados para as funções Rosenbrock e Beale.

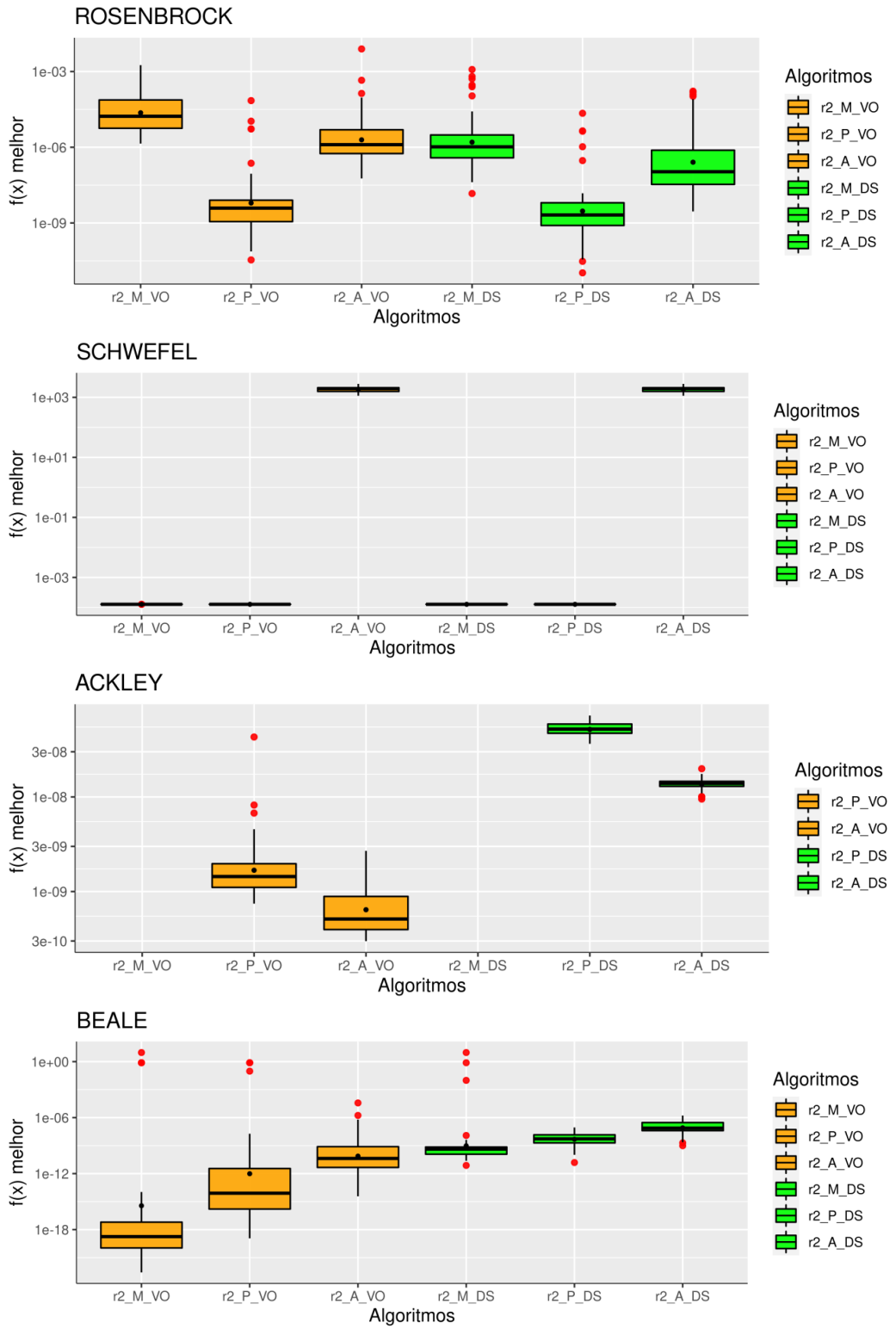
Uma análise estatística dos valores atingidos ao final das execuções do GEOreal2 foi realizada e foi gerado um boxplot para cada função comparando as seis implementações (Figura 5.10). Todos os boxplots estão na escala logarítmica.

Figura 5.10. Boxplots dos melhores valores obtidos ao final das 50 execuções para o GEOreal2.



(continua)

Figura 5.10. Conclusão.



Todos os gráficos estão em escala logarítmica e os nomes r2_M_VO, r2_P_VO, r2_A_VO, r2_M_DS, r2_P_DS e r2_A_DS representam, respectivamente, os algoritmos GEOreal2_M_VO, GEOreal2_P_VO, GEOreal2_A_VO, GEOreal2_M_DS, GEOreal2_P_DS e GEOreal2_A_DS.
Fonte: Produção do autor.

Os boxplots da Figura 5.10 corroboram com os resultados da Figura 5.9, onde é possível constatar que as diferentes maneiras de perturbar diversas vezes as variáveis de projeto em uma única iteração (diferentes marcadores nos gráficos da Figura 5.9) não possuem ganhos significativos exceto na função Beale, onde a variação DS foi muito superior do que a variação VO. É importante observar que nos boxplots das funções Griewangk, Rastrigin e Ackley, os algoritmos GEOreal2_M_VO e GEOreal2_M_DS atingiram o 0 em todas as execuções, não sendo representado nos boxplots.

Ao comparar o desempenho entre as formas de perturbar as variáveis utilizadas, avista-se um melhor desempenho da mutação P para todas as funções objetivo, confirmando que essa maneira de perturbar as variáveis é interessante. Inclusive, na função Schwefel a mutação P foi semelhante a mutação M. Nesta função, a diferença entre os resultados finais obtidos na busca utilizando mutação Multiplicativa e Porcentagem foi na décima casa decimal.

É possível analisar que todas as execuções GEOreal2 atingiram o ótimo na função Rastrigin, mesmo que os valores atingidos pelas versões DS tenham atingido 10^{-16}. Por fim, embora na função Beale os algoritmos do tipo VO tenham atingido um resultado muito superior quando considerado apenas a mediana, a presença de outliers influencia ao elevar a média aritmética.

Considerando os melhores desempenhos dos algoritmos que utilizaram a mutação P e a forma DS de variar os desvios padrões, um novo teste visando o aumento de desempenho desse algoritmo foi realizado. Esse teste visou modificar a primeira das P mutações para ser uma mutação com distribuição uniforme entre o intervalo de variação das variáveis, conforme a equação:

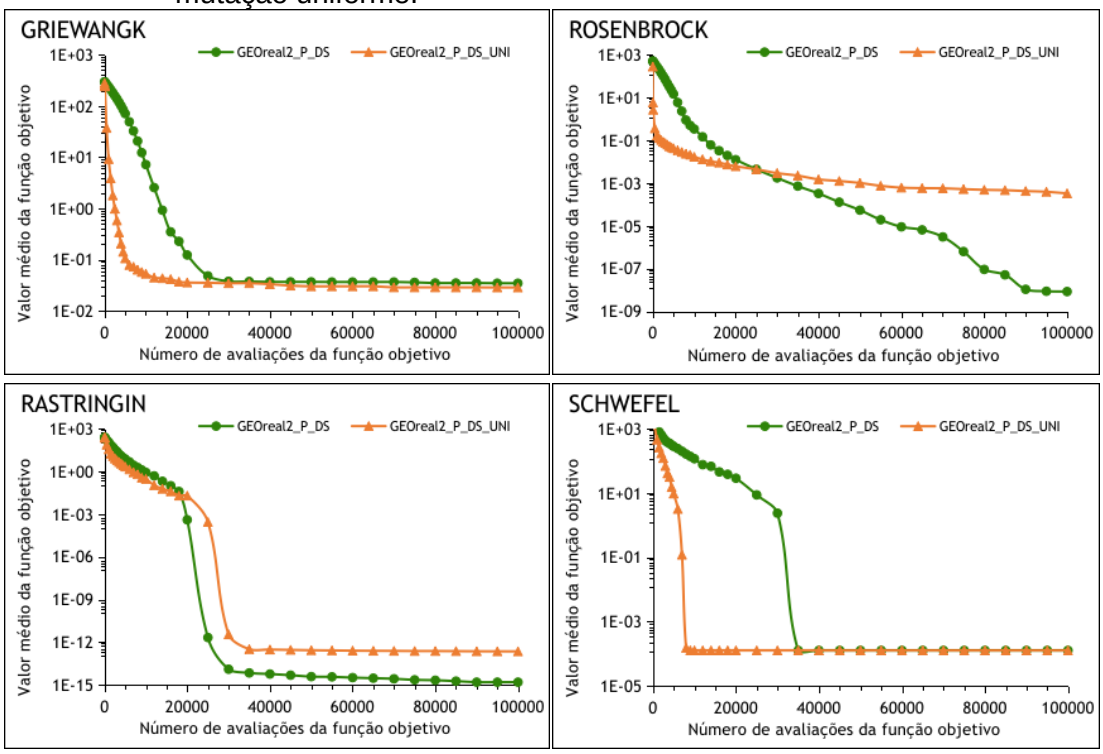
$$x_i = x^L + [0,1] \times (x^U - x^L). \tag{5.1}$$

Na Equação 5.1, x_i é o valor atual da variável, [0,1] é um valor aleatório da distribuição uniforme de probabilidade e x^U e x^L são, respectivamente, o limite superior e inferior da variável de projeto. A adição dessa perturbação uniforme faria com que uma das P mutações fosse qualquer valor dentro do intervalo de

variação de cada variável, podendo esse ser um valor muito maior do que a primeira perturbação que utiliza o σ1 apresentado na Tabela 5.2

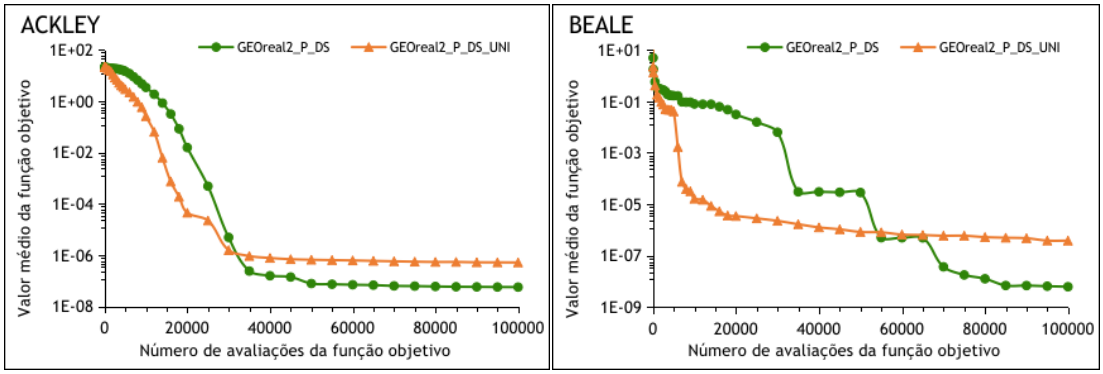
A nova versão gerada foi denominada GEOreal2_P_DS_UNI e ela foi comparada com a versão GEOreal2_P_DS, sendo esse desempenho apresentada na Figura 5.11.

Figura 5.11. Comparativo de desempenho da versão GEOreal2_P_DS com e sem a mutação uniforme.



(continua)

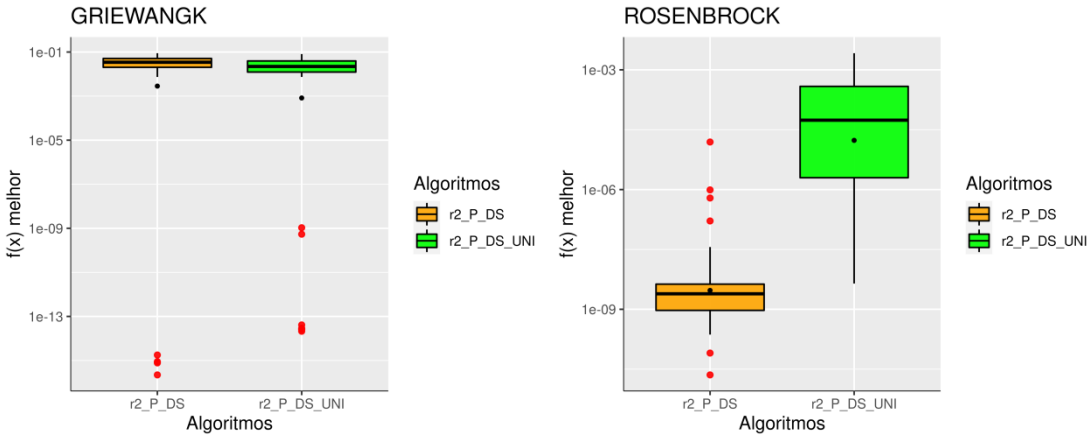
Figura 5.11. Conclusão.



Fonte: Produção do autor.

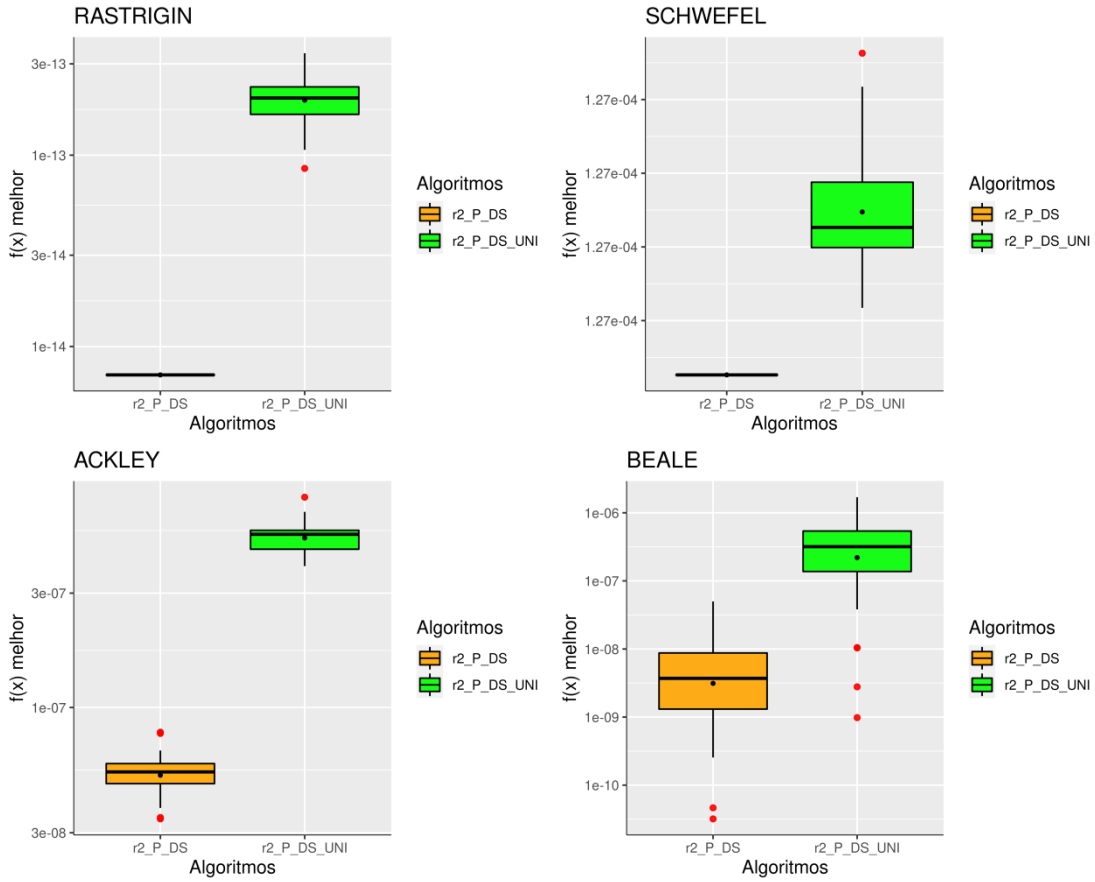
Os boxplots com os valores atingidos ao final das 50 buscas foram gerados para ambos algoritmos e são apresentados na escala logarítmica na Figura 5.12.

Figura 5.12. Boxplots dos melhores valores obtidos ao final das 50 execuções para os algoritmos GEOreal2_P_DS e GEOreal2_P_DS_UNI.



(continua)

Figura 5.12. Conclusão.



Todos os gráficos estão em escala logarítmica e os nomes r2_P_DS e r2_P_DS_UNI representam, respectivamente, os algoritmos GEOreal2_P_DS e GEOreal2_P_DS_UNI.

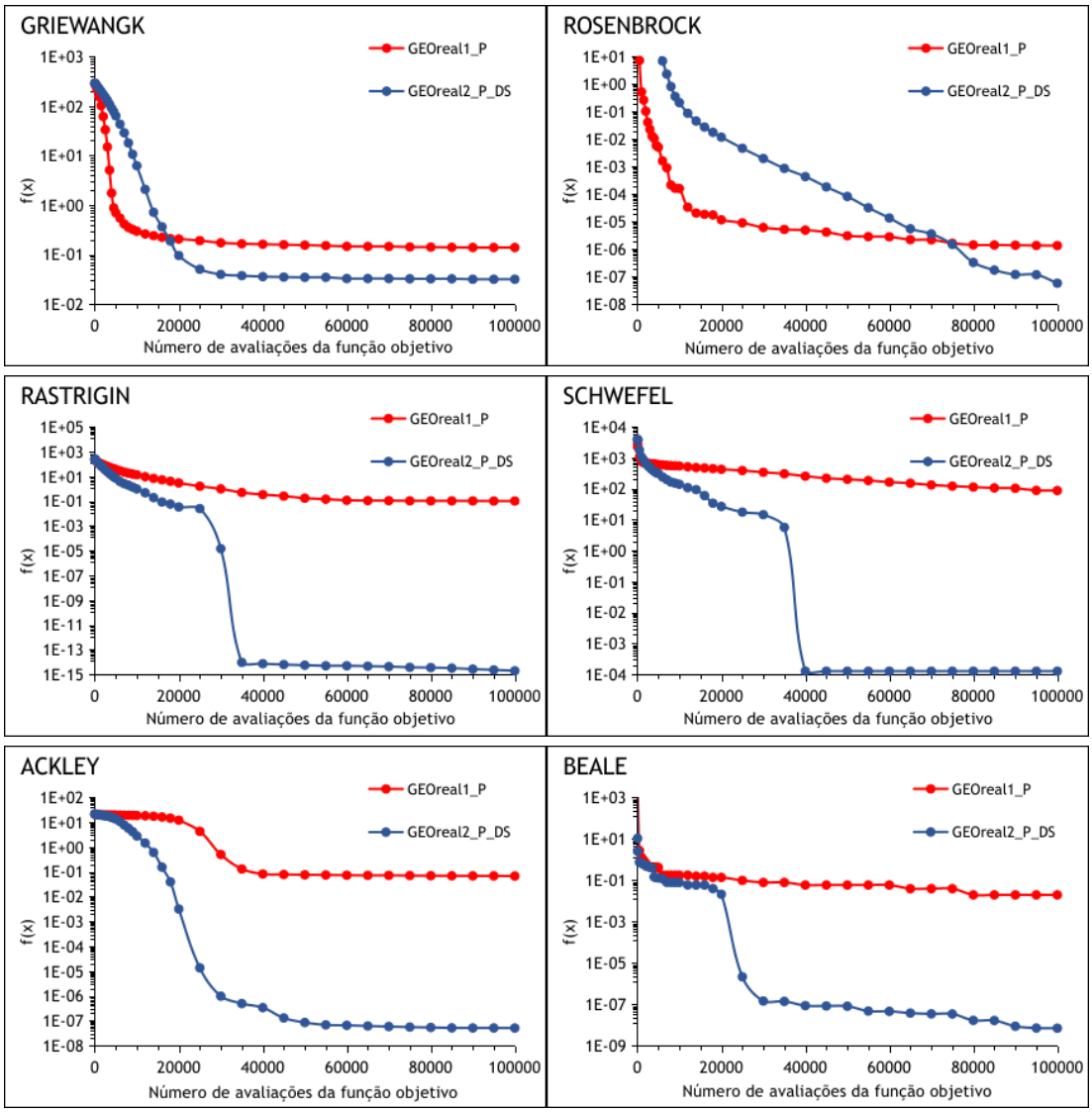
Fonte: Produção do autor.

Os desempenhos observados na Figura 5.11 juntamente com os boxplots apresentados na Figura 5.12 permitem observar que a adição de uma perturbação uniforme não melhorou significativamente o resultado. Apesar da melhor convergência para todas as funções objetivo para baixos valores de NFE, o valor final foi pior do que a implementação GEOreal2_P_DS. Com isso, é possível concluir que o algoritmo GEOreal2_P_DS foi o que obteve o melhor desempenho dentre os algoritmos GEOreal2.

Finalmente, foi feita uma comparação entre o melhor algoritmo GEOreal1 com o melhor GEOreal2 a fim de verificar qual das implementações é mais

vantajosa. Esse comparativo de desempenho foi realizado e é apresentado na Figura 5.13.

Figura 5.13. Comparativo de desempenho da melhor versão do GEOreal1 com a melhor versão do GEOreal2.

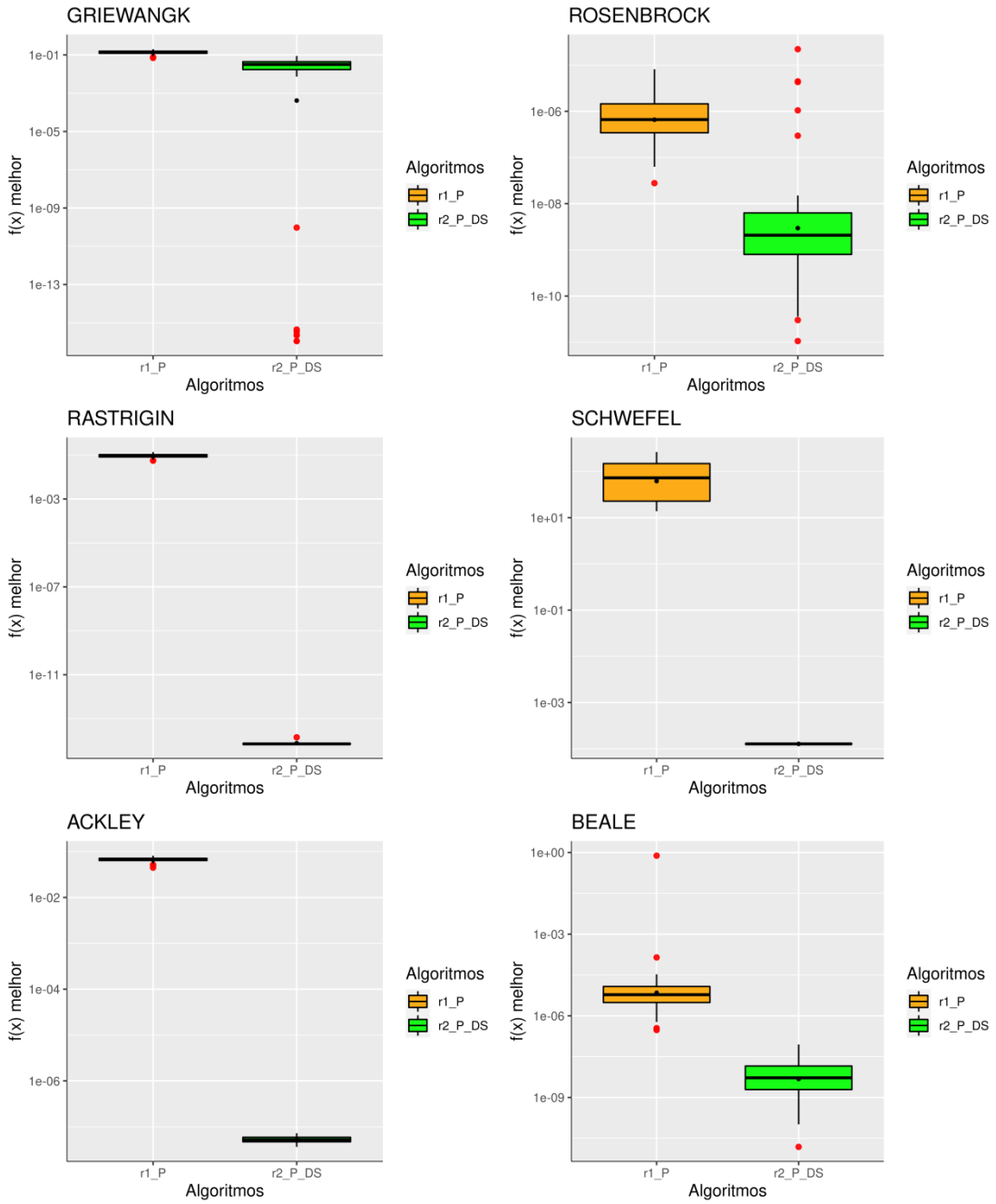


Fonte: Produção do autor.

Apesar da pior convergência nas funções Griewangk e Rosenbrock, o GEOreal2_P_DS atingiu o melhor desempenho para todas as funções teste, mostrando que o GEOreal2 possui melhor desempenho que o GEOreal1 para estas funções. Estes resultados são complementados por boxplots que foram gerados utilizando os valores atingidos ao final das 50 buscas para cada um

dos dois algoritmos. Estes resultados são apresentados na Figura 5.14 e os boxplots estão em escala logarítmica.

Figura 5.14. Boxplots dos melhores valores obtidos ao final das 50 execuções para os algoritmos GEOreal1_P e GEOreal2_P_DS.



Todos os gráficos estão em escala logarítmica e os nomes r1_P e r2_P_DS representam, respectivamente, os algoritmos GEOreal1_P e GEOreal2_P_DS.

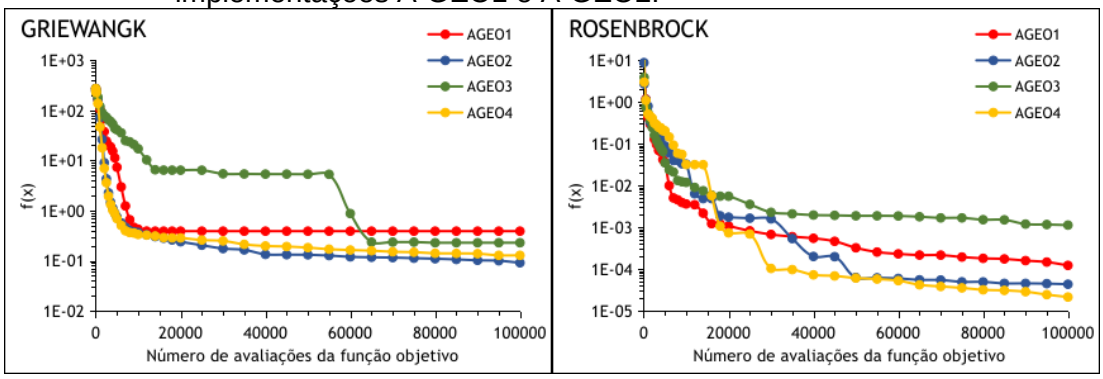
Fonte: Produção do autor.

Os resultados comprovam que realizar diversas mutações em cada variável de projeto a cada iteração é mais favorável (GEOreal2). O custo computacional e a quantidade de parâmetros livres é maior, mas o ganho de desempenho é muito superior, chegando a atingir uma diferença de oito ordens de grandeza nas funções Ackley e Beale. Por fim, é importante avaliar que a implementação GEOreal2_P_DS atingiu o ótimo na função Rastringin, sendo essa uma função onde x* = 0, o que mostra que a mutação P possui um bom potencial em encontrar soluções candidatas.

5.4 Avaliação do desempenho por meio da alteração do mecanismo adaptativo do τ

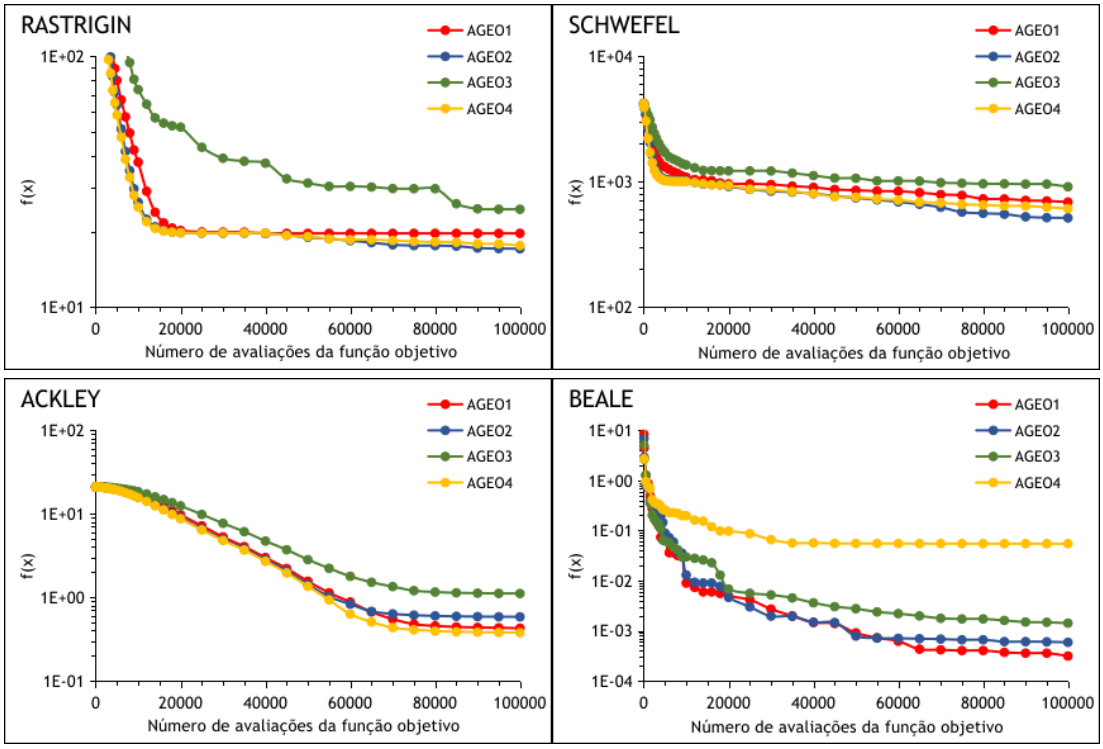
As duas novas modificações testadas visando alterar o mecanismo de adaptação do τ foram implementadas e seus desempenhos são apresentados na Figura 5.15. Essas implementações, denominadas A-GEO3 e A-GEO4, foram comparadas com as versões A-GEO1 e A-GEO2 e os gráficos mostram a média do melhor valor da função objetivo em função do NFE.

Figura 5.15. Desempenho das variações A-GEO3 e A-GEO4 comparadas às implementações A-GEO1 e A-GEO2.



(continua)

Figura 5.15. Conclusão.



Fonte: Produção do autor.

Os gráficos permitem observar que a versão A-GEO3 teve o pior desempenho para todas as funções teste exceto a Beale, onde foi superior somente ao A-GEO4. Isso indica que essa implementação não adiciona vantagens em termos de desempenho. Além disso, há um aumento de complexidade em sua implementação, visto que na sua regra é preciso realizar 3 testes e informações como o CoI da iteração anterior são necessários.

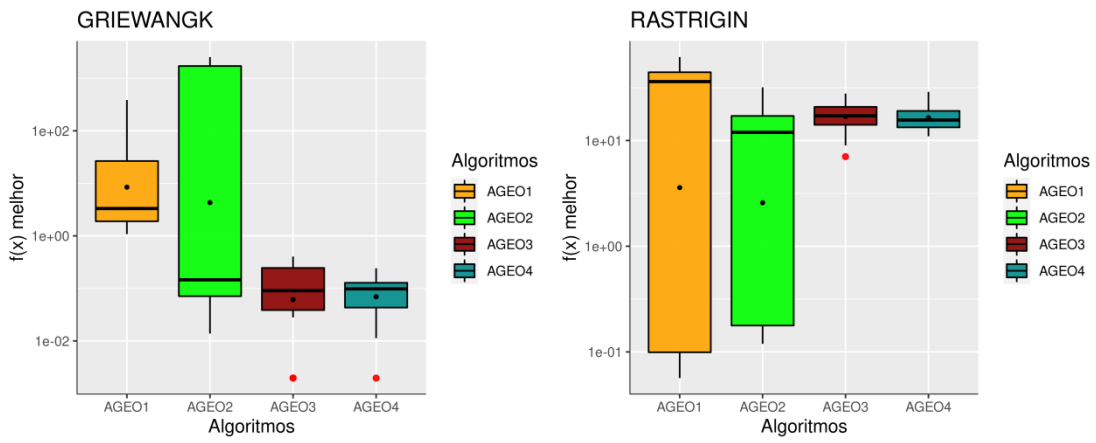
De modo geral, o desempenho do algoritmo A-GEO4 é semelhante ao algoritmo A-GEO2, exceto para a função Beale. Isso indica que aumentar o valor de τ sempre que há qualquer solução que melhore o valor da solução atual é interessante. Inclusive, para todas as funções exceto a Beale, essa implementação foi melhor que as versões A-GEO1 e A-GEO3. No A-GEO4, o valor de τ é aumentado sempre que o CoI for maior que 0, ou seja, sempre que um indivíduo da população melhora em relação a população de referência. Isso significa que quando ao menos um indivíduo melhorou, a busca se torna mais determinística. Isso difere do A-GEO2, onde o valor de τ é aumentado somente

quando o CoI da iteração atual é menor que o da anterior (menos indivíduos melhoraram em relação a iteração anterior). Visto que o A-GEO4 atingiu um desempenho semelhante ao A-GEO2 para a maioria das funções, este surge como uma alternativa ao mecanismo, pois nesse não é necessário armazenar durante a busca o CoI calculado na iteração anterior.

Durante a pesquisa, uma outra abordagem ainda foi testada, consistindo em utilizar como população de referência a solução do melhor indivíduo gerado após as L mutações, onde L é a quantidade de bits total da população. Essa implementação teve um desempenho muito ruim em comparação com o A-GEO2. Ao utilizar como população de referência a solução do melhor indivíduo gerado após as L mutações, o CoI é calculado como a razão entre a quantidade de bits que quando mutados provém uma solução melhor que a de referência (a referência aqui é a melhor solução dentre as L mutações) e a quantidade total de bits, então nunca haverá uma solução melhor que a melhor das L mutações. Isso faz com que o valor de CoI seja sempre 0 e o valor de τ seja resetado em todas as iterações, tornando a busca sempre estocástica e não equilibrando as características desejadas de exploração e aprimoramento.

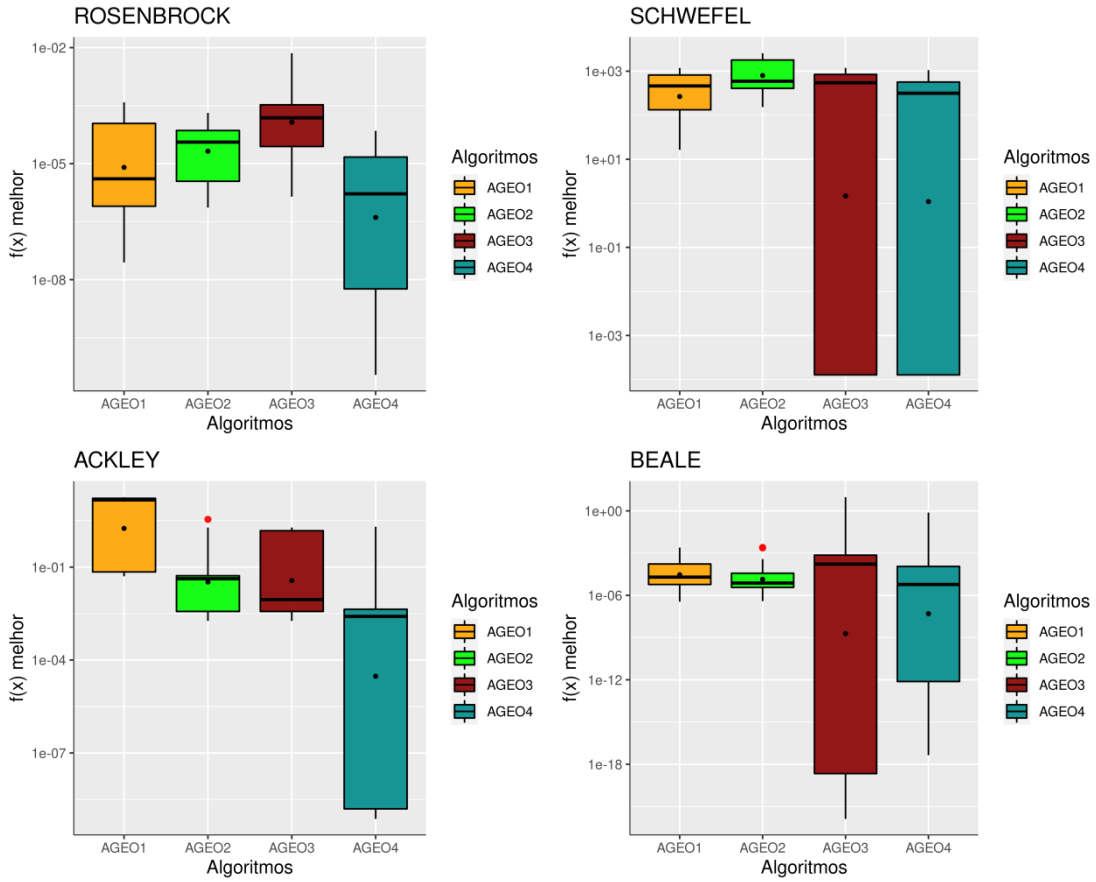
Esta subseção também apresenta uma análise estatística dos valores atingidos ao final das 50 execuções dos algoritmos A-GEO1, A-GEO2, A-GEO3 e A-GEO4 (Figura 5.16). Todos os boxplots estão na escala logarítmica.

Figura 5.16. Boxplots dos melhores valores obtidos ao final das 50 execuções para os algoritmos A-GEO1, A-GEO2, A-GEO3 e A-GEO4. GEOreal1.



(continua)

Figura 5.16. Conclusão.



Todos os gráficos estão em escala logarítmica.

Fonte: Produção do autor.

A partir dos boxplots, infere-se que o A-GEO2 e o A-GEO4 tiveram desempenhos semelhantes para todas as funções objetivo. Embora o A-GEO4 tenha atingido uma maior dispersão em funções como a Schwefel, Ackley, Beale, a mediana das duas implementações foi semelhante. Quando comparado utilizando a média aritmética, é possível observar uma diferença de desempenho na função Ackley, pois o A-GEO3 foi superior ao A-GEO1 e A-GEO2 na mediana, mas foi pior na média aritmética conforme apresentado na Figura 5.15, em razão de outliers.

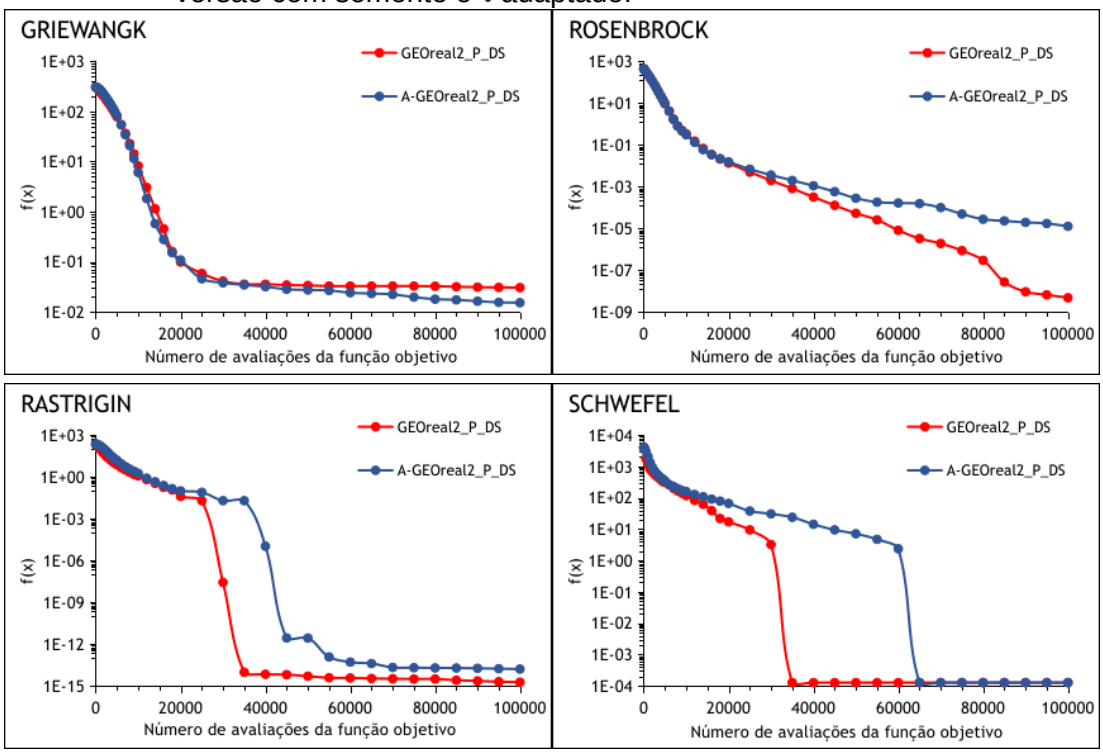
Ao fim das comparações de desempenho utilizando todas as versões adaptativas binárias, infere-se que a melhor solução de referência a ser utilizada no cálculo do CoI é a solução corrente, como utilizada nas versões A-

GEO2, A-GEO2var e A-GEO4. Ao comparar os novos indivíduos gerados pelas mutações durante a busca com a solução corrente, é possível observar se a busca está sendo aprimorada ou se é necessário explorar novos pontos do espaço de projeto através da reinicialização do τ, tornando o algoritmo mais estocástico.

5.5 Adaptação do GEO com codificação real (A-GEOreal)

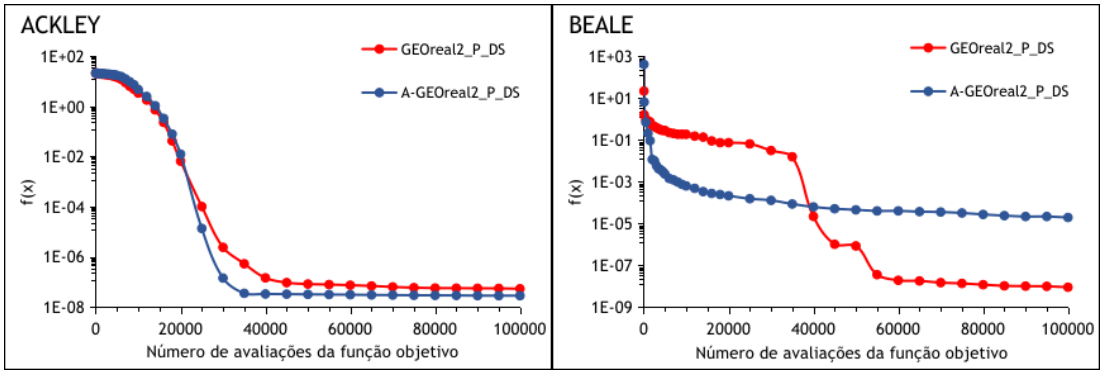
O comparativo de desempenho entre o A-GEOreal2_P_DS e a sua base GEOreal2_P_DS é apresentado na Figura 5.17. Os gráficos apresentam o valor médio da função objetivo em função do NFE para 50 execuções independentes.

Figura 5.17. Comparativo de desempenho da versão GEOreal2_P_DS e de sua versão com somente o τ adaptado.



(continua)

Figura 5.17. Conclusão.



Fonte: Produção do autor.

De forma geral, a versão com todos os parâmetros ajustados foi igual ou superior à versão com o τ adaptado para todas as funções teste. Apesar do melhor valor atingido pela versão adaptativa nas funções Griewangk e Ackley, é preciso levar em conta o desvio-padrão da média aritmética das execuções, o que permitiria concluir que os algoritmos empataram nestas funções. As maiores diferenças apresentadas na média aritmética do valor da função objetivo entre os dois algoritmos foram de cerca de 4 ordens de grandeza e ocorreram nas funções Rosenbrock e Beale. Essa última função foi a única função onde a versão adaptativa atingiu melhor convergência do que a versão ajustada.

Não foram gerados boxplots para estes resultados, visto que essa foi uma primeira implementação para testar a influência de tornar apenas um dos parâmetros adaptativo. Portanto, visando desenvolver uma versão completamente adaptativa deste algoritmo, é necessária uma análise mais detalhada sobre os outros 3 parâmetros livres a fim de fixá-los ou adaptá-los. É importante observar que ainda não existe nenhuma versão completamente adaptativa do A-GEO com codificação real.

No algoritmo A-GEO de codificação binária, o fato de ter tornado o GEO adaptativo melhorou o desempenho porque o mecanismo conseguiu controlar de forma eficiente a capacidade de exploração e aprimoramento da busca no único parâmetro livre que era o τ. Porém, nas versões com codificação real, há pelo menos mais um parâmetro livre além do τ. No caso da implementação que

mostrou melhores resultados para a codificação real, esse parâmetro denominado ρ é a porcentagem do intervalo de variação da variável e age diretamente na mutação das variáveis de projeto, permitindo grandes e pequenas alterações no valor das mesmas. Portanto, como estes dois parâmetros agem diretamente na capacidade de exploração e aprimoramento da busca, significa que eles possuem uma relação direta e um depende do outro para que a busca atinja um equilíbrio nessas capacidades. Na etapa de exploração (exploration), quando o valor de τ é baixo, pode ser interessante grandes valores para ρ, permitindo grandes saltos no espaço de projeto para fugir de mínimos locais. Já no aprimoramento da busca (exploitation), torna-se interessante pequenos valores de ρ, permitindo a busca dar pequenos saltos no espaço de projeto e explorar as soluções locais.

A etapa de ajuste de parâmetros de todas as versões do GEOreal2 foi apresentada na Tabela 5.2. Entretanto, como a análise atual visa estudar os parâmetros ρ1, P e s do GEOreal2_P_DS, o fragmento relacionado a esse algoritmo foi copiado e é apresentado na Tabela 5.3 para uma análise mais detalhada.

Tabela 5.3. Valores dos parâmetros que resultam no melhor desempenho do algoritmo GEOreal2_P_DS.

Parâmetro Livres	**Gri**	**Ras**	**Ack**	**Ros**	**Sch**	**Bea**
P: n. perturbações	10	10	10	5	10	5
s: Variação Pert.	10	10	10	2	10	10
ρ1: Porcentagem	1,0	10,0	10,0	0,1	50,0	50,0

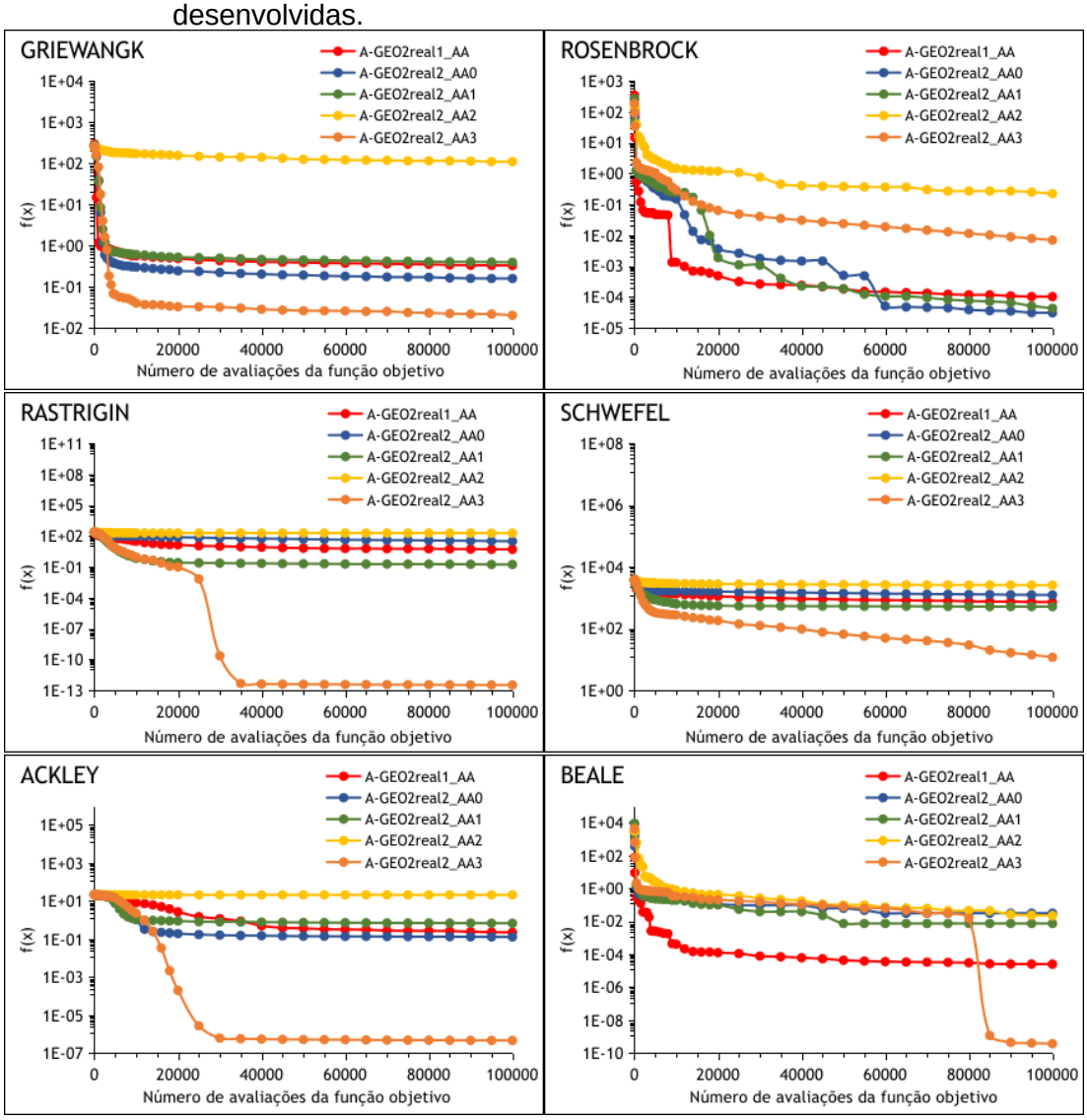
Fonte: Produção do autor.

Os valores ajustados dos parâmetros do GEOreal2_P_DS para cada função teste permitem encontrar alguns padrões. Primeiramente, observa-se que o valor do parâmetro P para todas as funções teste, exceto Rosenbrock e Beale, foi 10. Além disso, esse mesmo valor foi obtido no parâmetro s em todas as funções teste, exceto a Rosenbrock. Isso significa que é possível fixar esses

dois parâmetros em 10, fazendo com que o algoritmo realize 10 mutações e a cada uma delas vá dividindo a porcentagem ρ por 10. Essa decisão não favorece todos os tipos de função objetivo, como por exemplo a Rosenbrock. Porém, esses valores foram comuns para quase todas as funções teste.

Todas as 5 implementações auto-adaptativas foram utilizadas nas funções teste e a Figura 5.18 apresenta o comparativo de desempenho entre os algoritmos.

Figura 5.18. Comparativo de desempenho entre as 5 versões auto-adaptativas desenvolvidas.



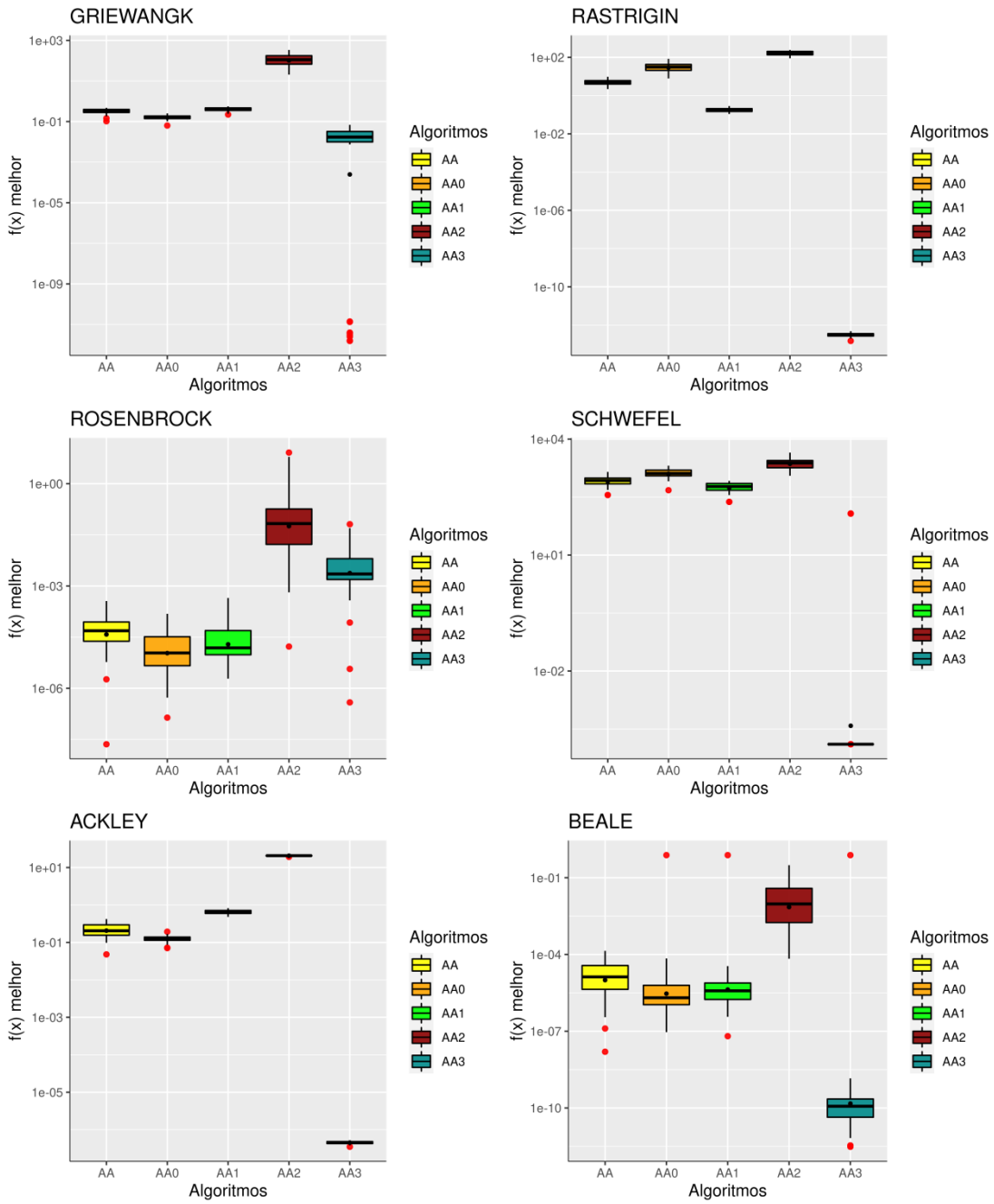
Fonte: Produção do autor.

Dentre todas as cinco versões avaliadas, a implementação A-GEO2real2_AA3 foi a que mais se destacou, chegando a atingir uma diferença de no mínimo 6 ordens de grandeza para a segunda melhor versão nas funções Rastrigin e Ackley. Essa versão foi a que atingiu o melhor desempenho para todas as funções, exceto a Rosenbrock, mostrando que essa maneira de auto-adaptar o parâmetro ρ1 é vantajosa. Ao buscar perturbar todas as variáveis ao mesmo tempo com um pequeno valor para ρ1' no momento de calcular suas adaptabilidades, essa abordagem se mostrou ser mais interessante do que se fosse calcular as adaptabilidades de ρ1 com altos valores de ρ1'.

A versão A-GEO2real2_AA2 obteve o pior desempenho para todas as funções teste. Isso possivelmente ocorreu devido ao valor de ρ confirmado para a próxima iteração sempre ser um valor pequeno e que vai sendo dividido 10 vezes por 10. Isso faz com que seu valor atinja valores muito baixos que sempre vão sendo divididos por 10 a cada iteração. Durante a execução, esses valores para ρ atingiram 10^{-21}, fazendo com que a mutação fosse tão pequena que a representação de ponto flutuante da linguagem de programação utilizada não conseguisse representar. Essa abordagem difere da versão AA3 que parte de um ρ1' e gera os valores de ρ' ao multiplicar ρ1' 4 vezes por 10 e dividir 4 vezes por 10, fazendo com que grandes e pequenas porcentagens sejam utilizadas em uma única iteração, ao invés de somente realizar divisões por 10.

A Figura 5.19 apresenta os boxplots dos valores atingidos ao final das 50 execuções dos algoritmos auto-adaptativos. Todas as escalas do eixo y dos boxplots estão na escala logarítmica.

Figura 5.19. Boxplots dos melhores valores obtidos ao final das 50 execuções para os algoritmos auto-adaptativos.



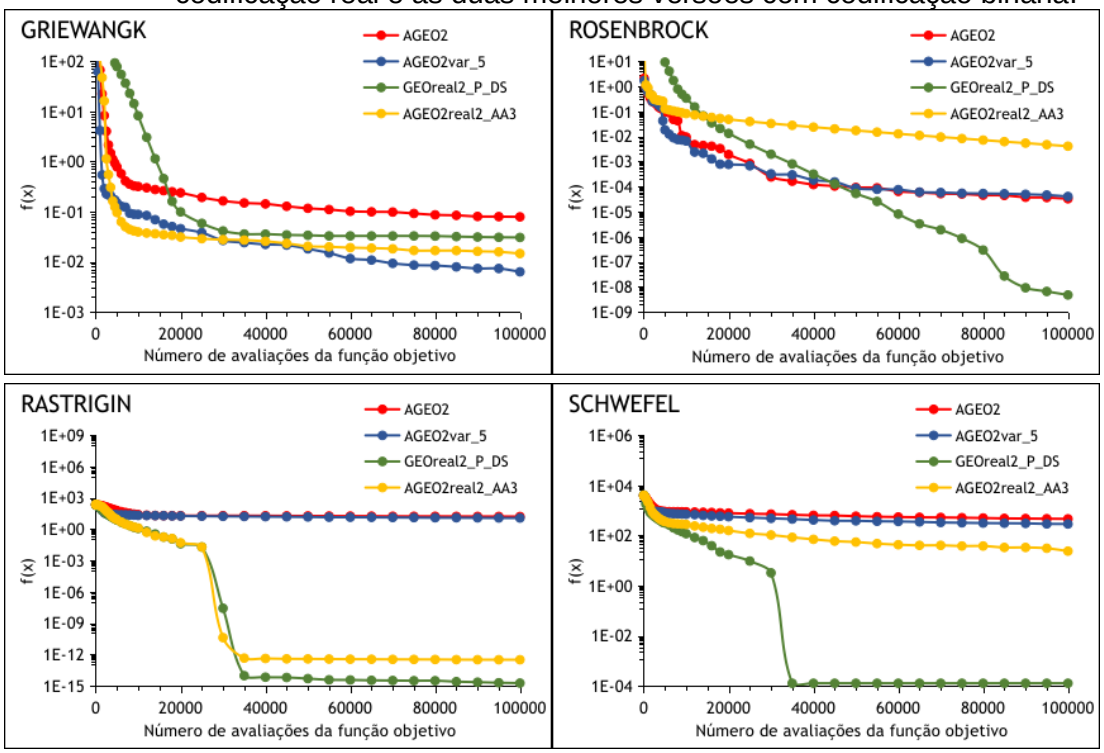
Na figura, todos os gráficos estão em escala logarítmica e os nomes AA, AA0, AA1, AA2 e AA3 representam, respectivamente, os algoritmos A-GEO2real1_AA, A-GEO2real2_AA0, A-GEO2real2_AA1, A-GEO2real2_AA2 e A-GEO2real2_AA3.

Fonte: Produção do autor.

Ao analisar tanto a Figura 5.18 quanto a 5.19, observa-se que a implementação A-GEO2real1_AA foi a terceira melhor implementação para todas as funções teste. Essa versão faz somente uma mutação em cada variável de projeto a cada iteração e difere das demais que realizam várias mutações a cada iteração. Nas funções Rosenbrock e Beale, essa versão foi a que mais se destacou em termos de convergência.

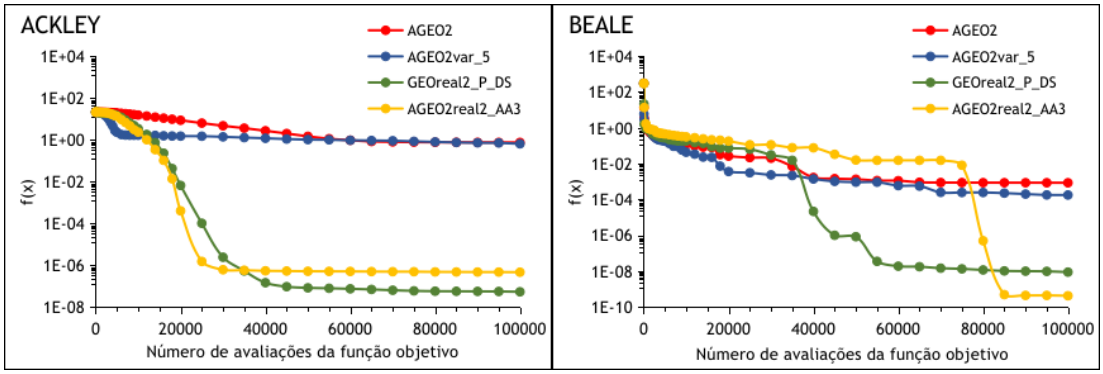
A Figura 5.19 confirma os resultados apresentados na Figura 5.18, mostrando que o algoritmo A-GEO2real2_AA3 atingiu o melhor desempenho para a maioria das funções teste. Portanto, ao constatar que esta foi a melhor versão auto-adaptativa, a próxima etapa consistiu em compará-la com a melhor versão real até o momento que foi a implementação totalmente ajustada GEOreal2_P_DS. Juntamente, são apresentados os resultados da melhor versão binária até o momento, A-GEO2var_5 e a versão base A-GEO2. O comparativo de desempenho entre as versões é apresentado na Figura 5.20.

Figura 5.20. Comparativo de desempenho entre as duas melhores versões com codificação real e as duas melhores versões com codificação binária.



(continua)

Figura 5.20. Conclusão.

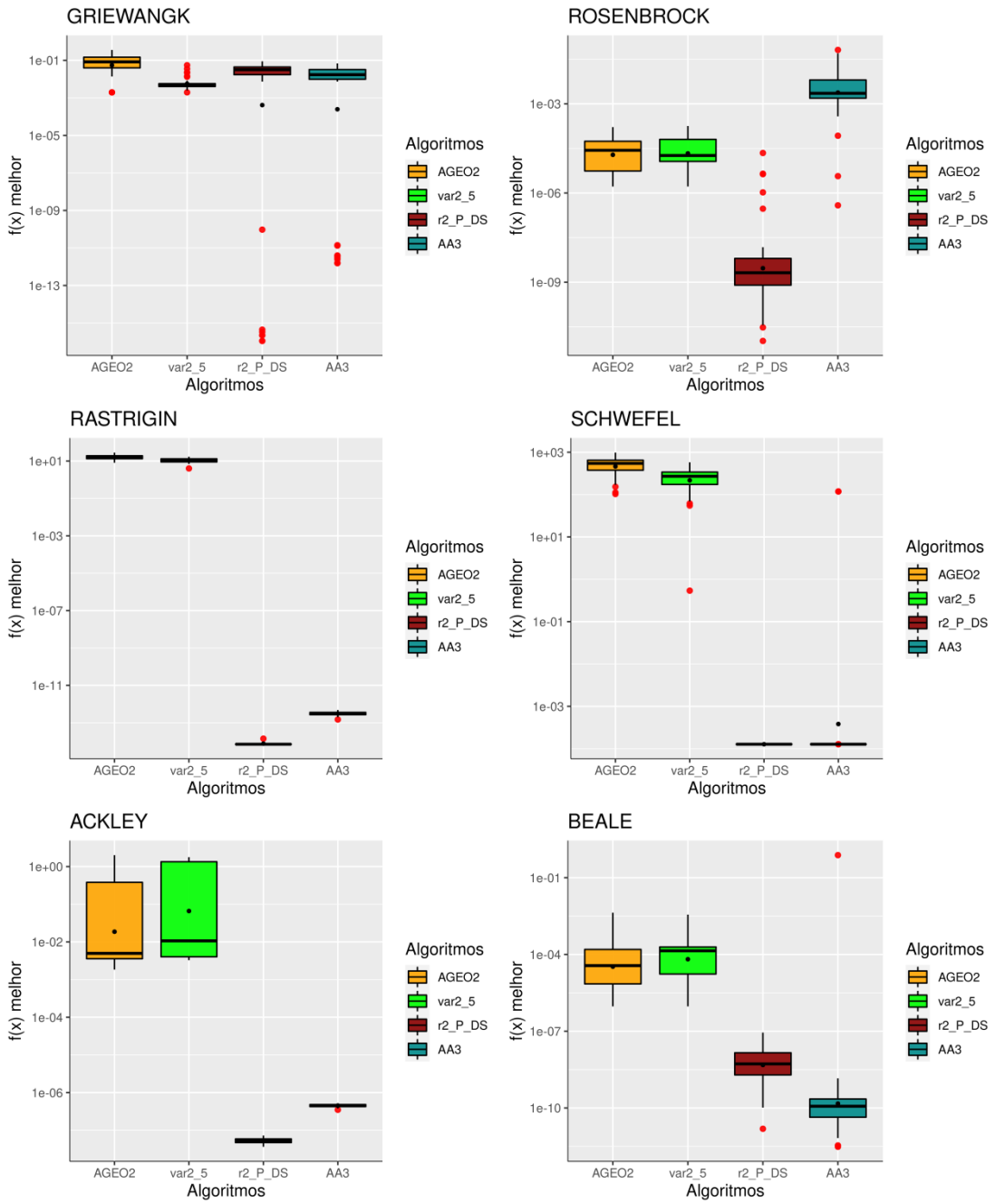


Fonte: Produção do autor.

Em todas as funções teste, há uma diferença significativa entre as implementações com codificação real e as implementações com codificação binária. Para a maioria das funções, os dois piores desempenhos foram obtidos com as versões que utilizam codificação binária e os dois melhores desempenhos com as versões que utilizam codificação real. Isso só não ocorreu na função Rosenbrock, onde o desempenho da implementação real auto-adaptativa foi pior que as implementações binárias, e na função Griewangk onde a versão A-GEO2var_5 foi superior às versões com codificação real.

Os boxplots com os valores atingidos ao final das 50 buscas foram gerados para os algoritmos A-GEO2, A-GEO2var_5, GEOreal2_P_DS e A-GEO2real2_AA3 e são apresentados na Figura 5.21. Os boxplots estão em escala logarítmica.

Figura 5.21. Boxplots dos melhores valores obtidos ao final das 50 execuções para os melhores algoritmos obtidos no trabalho.



Na figura, todos os gráficos estão em escala logarítmica e os nomes AGEO2, var2_5, r2_P_DS e AA3 representam, respectivamente, os algoritmos A-GEO2, A-GEO2var_5, GEOreal2_P_DS e A-GEO2real2_AA3.

Fonte: Produção do autor.

Ao analisar tanto a Figura 5.20 quanto a Figura 5.21, observa-se que, apesar da versão real semi auto-adaptativa (A-GEO2real2_AA3) atingir uma mediana melhor nas funções Griewangk e Beale, foi a versão com os parâmetros ajustados (GEOreal2_P_DS) que obteve o melhor resultado médio para todas as outras cinco funções teste. Além disso, para todas as funções, exceto a Rosenbrock, as implementações com codificação real foram muito superiores quando comparadas às implmementações com codificação binária.

O fato de ajustar os parâmetros faz com que muitas combinações de parâmetros sejam testadas, cada uma avaliando a média de várias execuções independentes, e muito mais avaliações da função objetivo sejam realizadas, aumentando o custo computacional. Esse caso torna-se muito importante de ser levado em conta, principalmente quando a função objetivo já é custosa computacionalmente, como em aplicações reais. Apesar do melhor desempenho do GEOreal2_P_DS, durante sua etapa de ajuste de parâmetros foram testadas 10 variações para o τ, 2 variações para o P, 2 variações para o s e 4 variações para o $\rho 1$, totalizando 160 combinações possíveis para os 4 parâmetros livres deste algoritmo. Para cada combinação, o algoritmo foi executado por 30 vezes com um critério de parada até 100000 NFEs e a média do melhor valor da função foi obtida. Isso significa que ao total foram realizadas 480 milhões de avaliações da função objetivo somente para encontrar a combinação de parâmetros que resultasse no melhor desempenho. Esse número pode ser reduzido ao testar menos combinações de parâmetros ou definir um critério de parada menor de NFE. Por outro lado, é importante observar que a quantidade de combinações de parâmetros testadas é proporcional ao desempenho, ou seja, quanto mais combinações são testadas mais fácil é encontrar uma combinação que extrai o melhor possível do algoritmo.

A comparação de desempenho da Figura 5.20 foi realizada obtendo a média de 50 execuções independentes, então mesmo executando a versão semi auto-adaptativa por 50 vezes com o critério de parada sendo 100.000 NFEs, houve um total de 5 milhões de NFEs, um valor que equivale a aproximadamente 1% da quantidade de avaliações realizadas somente na etapa de ajuste de

parâmetros do GEOreal2_P_DS. Portanto, o próximo experimento realizado buscou executar A-GEO2real2_AA3 até 100 milhões de NFEs e comparar o resultado final com o GEOreal2_P_DS executado até 100.000 NFEs. Com isso, buscava-se verificar se a versão ajustada continuaria superior à versão semi auto-adaptativa para a maioria das funções. Esse desempenho é apresentado na Tabela 5.4.

Tabela 5.4. Comparativo de desempenho ao executar o GEOreal2_P_DS até 100 mil NFEs e o A-GEO2real2_AA3 até 100 milhões de NFEs.

Algoritmo	**GRI**	**RAS**	**ACK**	**ROS**	**SCH**	**BEA**
GEOreal2_P_DS	3,07E-02	**1,28E-15**	**4,92E-08**	4,09E-08	**1,27E-04**	5,31E-08
A-GEO2real2_AA3	**6,77E-13**	8,53E-14	3,04E-07	**2,22E-09**	**1,27E-04**	**1,22E-14**

Fonte: Produção do autor.

Para 4 das 6 funções teste (Griewangk, Rosenbrock, Schwefel e Beale), a versão semi auto-adaptativa atingiu um desempenho igual ou superior a versão ajustada, principalmente nas funções Griewangk e Beale, onde a diferença foi de no mínimo 6 ordens de grandeza. Nas demais funções, o desempenho foi inferior, mas ainda um desempenho significante quando levado em conta a quantidade de NFEs utilizados na etapa de ajuste de parâmetros do GEOreal2_P_DS.

O fato da implementação auto-adaptativa não possuir parâmetros livres reduz não só a quantidade de NFEs necessárias, quanto reduz o esforço de precisar ajustar os parâmetros. Na etapa de ajuste de parâmetros, é preciso desenvolver ou utilizar uma lógica que permita variar todas as combinações de parâmetros e obter qual delas resulta no melhor desempenho, o que torna a esta etapa não só custosa computacionalmente quanto trabalhosa. Ao considerar que na versão auto-adaptativa não é necessário realizar ajuste de parâmetros e que esta atingiu um desempenho competitivo quando comparada a versão com 4 parâmetros livres, é possível concluir que esta versão é

amigável ao usuário e torna-se mais interessante na busca pela automatização da criação de soluções em projetos complexos, como projetos de sistemas espaciais.

6 COMPARAÇÃO DE DESEMPENHO EM UM CONJUNTO DE FUNÇÕES DO IEEE CEC

A pesquisa de algoritmos de otimização para problemas monoobjetivos sem restrições é a base para o desenvolvimento de algoritmos mais complexos, como algoritmos multi-objetivos e para problemas com restrições (AWAD et al., 2017). Ao longo do tempo, vários concursos para testar o desempenho de algoritmos de otimização foram organizados como parte da conferência IEEE Congress on Evolutionary Computation (CEC). Estas competições têm o objetivo de comparar anualmente os algoritmos de pesquisa estocástica de última geração e incluem vários tipos de conjuntos de problemas de benchmark, como por exemplo otimização mono-objetiva, ruidosa, multiobjetiva e restrita (FISTER et al., 2021). Todos os novos algoritmos selecionados para a competição são testados em um conjunto de funções de benchmark monoobjetivas que podem ser transformadas em dinâmicas, composição de nichos, computacionalmente caras e muitas outras classes de problemas (AWAD et al., 2017).

Os Concursos da CEC denominados Real-Parameter Single Objective Optimization foram organizados em diferentes anos utilizando quatro diferentes sets de funções. Em 2005, o set de funções descrito por Suganthan et al. (2005) foi utilizado e em 2013 o set utilizado foi o de Liang et al. (2013a). Enquanto que em 2014 e 2016 a competição foi realizada utilizando o mesmo set de funções, descrito por Liang et al. (2013b), em 2017 e 2018 as competições utilizaram o set de funções descrito por Awad et al. (2017). Posteriormente, este concurso foi renomeado para a 100-Digit Challenge Special Session and the Competition on Single Objective Numerical Optimization em 2019 e 2020.

No presente trabalho, um comparativo de desempenho entre os algoritmos foi realizado utilizando o conjunto de funções CEC 2017 apresentado resumidamente na Tabela 5.2. O fato deste conjunto possuir funções com diversas características morfológicas pertimiu avaliar de forma mais ampla o desempenho dos algoritmos A-GEO2var_5 e A-GEO2real2_AA3

desenvolvidos. Estes dois algoritmos foram comparados com o algoritmo que obteve o melhor desempenho na competição CEC 2017, denominado Effective Butterfly Optimizer With Covariance Matrix Adapted Retreat Phase (EBOwithCMAR). Este é um algoritmo evolutivo baseado no comportamento de borboletas que utiliza uma fase de matriz de covariância para melhorar sua capacidade de busca local (KUMAR; MISRA; SINGH, 2017). O desempenho dos algoritmos também foi comparado com o A-GEO2, que é a melhor versão binária adaptativa do GEO desenvolvida no trabalho de Barroca (2019).

A Tabela 6.1 apresenta o comparativo de desempenho entre os algoritmos utilizando 10 variáveis de projeto para cada problema. Nela, são apresentadas a média atingida pelo algoritmo ao final das 50 execuções e o ranqueamento, onde 1 significa o algoritmo que obteve o melhor desempenho dentre os avaliados e 4 significa o algoritmo que obteve o pior desempenho. Ao fim, foi computada a média do ranqueamento, bem como a quantidade de vitórias obtidas para cada algoritmo. É importante observar que neste Capitulo não são gerados boxplots para uma análise estatística, pois não estão disponíveis os resultados finais de todas as execuções do EBOwithCMAR para cada função.

Tabela 6.1. Comparativo de desempenho entre os 4 algoritmos para todas as funções CEC utilizando dimensão D = 10.

D = 10								
Nro. Função	**AGEO2**		**AGEO2var_5**		**AGEO2real2_AA3**		**EBOwithCMAR**	
	Média	**Ordem**	**Média**	**Ordem**	**Média**	**Ordem**	**Média**	**Ordem**
1	7,77E+8	4	2,09E+8	3	2,17E+4	2	**0,00E+0**	1
2	-	-	-	-	-	-	-	-
3	2,77E+4	3	1,83E+4	2	2,21E+5	4	**0,00E+0**	1
4	4,71E+1	4	9,98E+0	3	2,97E+0	2	**0,00E+0**	1
5	1,97E+1	4	1,31E+1	2	1,76E+1	3	**0,00E+0**	1
6	5,11E+0	4	1,30E+0	3	2,01E-5	2	**0,00E+0**	1
7	5,61E+1	4	3,77E+1	3	3,08E+1	2	**1,06E+1**	1
8	2,84E+1	4	1,71E+1	2	2,19E+1	3	**0,00E+0**	1
9	2,03E+2	4	1,55E+1	2	1,11E+2	3	**0,00E+0**	1
10	7,38E+2	4	4,95E+2	2	5,85E+2	3	**3,72E+1**	1
11	3,78E+3	4	2,17E+2	3	1,41E+1	2	**0,00E+0**	1

(continua)

Tabela 6.1. Conclusão.

D = 10								
Nro. Função	**AGEO2**		**AGEO2var_5**		**AGEO2real2_AA3**		**EBOwithCMAR**	
	Média	**Ordem**	**Média**	**Ordem**	**Média**	**Ordem**	**Média**	**Ordem**
12	3,59E+7	4	8,48E+5	3	1,87E+5	2	**9,02E+1**	1
13	2,99E+7	4	1,58E+5	3	3,91E+4	2	**2,17E+0**	1
14	2,78E+7	4	2,47E+2	2	5,00E+4	3	**6,05E-2**	1
15	1,40E+6	4	3,37E+2	2	6,15E+4	3	**1,09E-1**	1
16	2,80E+2	3	7,90E+1	2	3,15E+2	4	**4,17E-1**	1
17	2,36E+2	4	2,85E+1	2	6,90E+1	3	**1,47E-1**	1
18	2,38E+6	4	3,51E+3	2	7,65E+4	3	**7,00E-1**	1
19	5,00E+6	4	1,87E+2	2	5,86E+4	3	**1,50E-2**	1
20	2,50E+2	4	2,67E+1	3	7,00E+0	2	**1,47E-1**	1
21	2,13E+2	4	**2,12E+1**	1	2,09E+2	3	1,14E+2	2
22	5,61E+2	3	1,22E+2	2	6,40E+2	4	**9,85E+1**	1
23	3,32E+2	3	3,12E+2	2	3,35E+2	4	**3,00E+2**	1
24	3,16E+2	2	3,40E+2	4	3,20E+2	3	**1,66E+2**	1
25	4,89E+2	4	4,28E+2	3	**3,99E+2**	1	4,12E+2	2
26	8,25E+2	3	4,14E+2	2	1,08E+3	4	**2,65E+2**	1
27	4,19E+2	3	3,98E+2	2	4,23E+2	4	**3,92E+2**	1
28	5,94E+2	4	4,67E+2	2	5,19E+2	3	**3,07E+2**	1
29	3,35E+2	3	3,14E+2	2	3,87E+2	4	**2,31E+2**	1
30	8,06E+5	4	7,26E+5	3	9,23E+4	2	**4,07E+2**	1
MÉDIA		**3,69**		**2.38**		**2,86**		**1,07**
VITÓRIAS	**0**		**1**		**1**		**27**	

Fonte: Produção do autor.

Observa-se pela média do ranqueamento que tanto a melhor versão binária A-GEO2var_5 quanto a melhor versão real A-GEO2real2_AA3 superaram a primeira versão binária do A-GEO. O EBOwithCMAR foi o algoritmo que atingiu o melhor desempenho tanto na média do ranqueamento quanto na quantidade de vitórias para a maioria das funções, vencendo em mais de 90% dos problemas. Apesar do A-GEO2 não ter sido o melhor algoritmo para nenhuma das funções, as versões A-GEO2var_5 e A-GEO2real2_AA3 venceram em pelo menos uma função.

Nesta comparação, o A-GEO2var_5 foi o segundo melhor algoritmo em termos de desempenho, perdendo apenas para o algoritmo EBOwithCMAR. O ranqueamento permite analisar que ele ficou 17 vezes em 2º lugar e 10 vezes

em 3º lugar, enquanto que a verãso auto-adaptativa com codificação real ficou 9 vezes em 2º lugar e 12 vezes em 3º lugar.

A Tabela 6.2 apresenta o comparativo de desempenho entre os algoritmos utilizando 30 variáveis de projeto para cada problema. Nela, são apresentadas a média atingida pelo algoritmo ao final das 51 execuções e o ranqueamento, onde 1 significa o algoritmo que obteve o melhor desempenho dentre estes e 4 significa o pior desempenho. Ao fim, foi computada a média do ranqueamento, bem como a quantidade de vitórias obtidas para cada algoritmo.

Tabela 6.2. Comparativo de desempenho entre os três algoritmos para todas as funções CEC utilizando dimensão D = 30.

D = 30								
Nro. Função	**AGEO2**		**AGEO2var_5**		**AGEO2real2_AA3**		**EBOwithCMAR**	
	Média	**Ordem**	**Média**	**Ordem**	**Média**	**Ordem**	**Média**	**Ordem**
1	4,22E+9	4	1,31E+9	3	2,13E+4	2	**0,00E+0**	1
2	-	-	-	-	-	-	-	-
3	1,95E+5	3	1,65E+5	2	3,38E+6	4	**0,00E+0**	1
4	3,80E+2	4	1,88E+2	3	**4,72E+1**	1	5,65E+1	2
5	1,53E+2	3	9,55E+1	2	1,54E+2	4	**2,78E+0**	1
6	1,06E+1	4	2,97E+0	3	5,93E-7	2	**0,00E+0**	1
7	2,80E+2	4	1,87E+2	3	1,75E+2	2	**3,35E+1**	1
8	1,51E+2	3	9,83E+1	2	1,71E+2	4	**2,02E+0**	1
9	6,08E+3	4	3,80E+2	2	4,98E+3	3	**0,00E+0**	1
10	3,63E+3	3	3,52E+3	2	3,44E+3	2	**1,41E+3**	1
11	1,19E+4	4	3,15E+3	3	1,17E+2	2	**4,49E+0**	1
12	6,58E+8	4	1,48E+8	3	1,95E+7	2	**4,63E+2**	1
13	1,52E+9	4	3,54E+7	3	6,95E+4	2	**1,49E+1**	1
14	9,11E+6	4	1,38E+6	2	1,46E+6	3	**2,19E+1**	1
15	4,14E+8	4	4,69E+6	3	3,74E+4	2	**3,69E+0**	1
16	1,46E+3	4	1,04E+3	2	1,39E+3	3	**4,26E+1**	1
17	8,01E+2	3	3,80E+2	2	8,15E+2	4	**2,98E+1**	1
18	1,10E+7	4	9,28E+6	3	4,94E+6	2	**2,21E+1**	1
19	3,65E+8	4	1,19E+7	3	2,43E+4	2	**8,04E+0**	1
20	7,50E+2	3	4,23E+2	2	8,36E+2	4	**3,57E+1**	1
21	3,46E+2	3	3,00E+2	2	3,60E+2	4	**1,99E+2**	1
22	4,01E+3	3	4,40E+2	2	4,21E+3	4	**1,00E+2**	1
23	5,08E+2	3	4,42E+2	2	5,17E+2	4	**3,51E+2**	1
24	7,21E+2	4	5,92E+2	2	6,25E+2	3	**4,18E+2**	1

(continua)

Tabela 6.2. Conclusão.

D = 30								
Nro. Função	**AGEO2**		**AGEO2var_5**		**AGEO2real2_AA3**		**EBOwithCMAR**	
	Média	**Ordem**	**Média**	**Ordem**	**Média**	**Ordem**	**Média**	**Ordem**
25	6,74E+2	4	4,91E+2	3	3,94E+2	2	**3,87E+2**	1
26	2,59E+3	3	2,09E+3	2	3,08E+3	4	**5,37E+2**	1
27	5,78E+2	4	5,51E+2	3	5,48E+2	2	**5,02E+2**	1
28	8,84E+2	4	5,92E+2	3	3,42E+2	2	**3,08E+2**	1
29	1,10E+3	3	1,17E+3	4	1,09E+3	2	**4,33E+2**	1
30	8,08E+7	4	2,54E+6	3	1,95E+4	2	**1,99E+3**	1
MÉDIA		**3.62**		**2.55**		**2.72**		**1.03**
VITÓRIAS	**0**		**0**		**1**		**28**	

Fonte: Produção do autor.

Ao utilizar dimensão 30, observa-se que o desempenho do A-GEO2real2_AA3 aumentou, uma vez que a média do seu ranqueamento diminuiu e a do A-GEO2var_5 aumentou. Todavia, a versão auto-adaptativa real foi a única que venceu o EBOwithCMAR em uma das funções, obtendo também 15 segundos lugares no ranqueamento.

Tanto para dimensão D=10 quanto dimensão D=30, a média do ranqueamento e a quantidade de vitórias permitiu observar que as duas implementações ainda não possuem capacidade de competir diretamente com algoritmos de alto desempenho como o EBOwithCMAR. Apesar do desempenho atingido já melhorar quando comparado ao A-GEO2 que foi o ponto inicial deste trabalho, o EBOwithCMAR venceu em mais de 90% dos problemas, mostrando ser um algoritmo muito eficiente na busca pelo ótimo. Portanto, ainda há muita pesquisa a ser desenvolvida visando melhorar o mecanismo do GEO adaptativo.

7 APLICAÇÃO EM UM PROBLEMA DE OTIMIZAÇÃO DE SISTEMA ESPACIAL

No intuito de apresentar os potenciais ganhos na utilização dos melhores algoritmos desenvolvidos no presente trabalho em um problema de otimização de sistema espacial na fase conceitual de uma missão espacial, utilizou-se o cenário descrito em Chagas et al. (2015), que foi o mesmo utilizado por Barroca (2019) como parte da avaliação de desempenho do A-GEO.

O problema consiste em modelar um satélite de observação da Terra e encontrar a órbita heliosícrona que minimize sua massa total. Em seu trabalho, Chagas (2015) apresenta que quanto menor a altitude da órbita de um satélite deste tipo, mais simples é o projeto da câmera devido à menor distância entre o equipamento e o alvo. Porém, quanto menor a altitude da órbita, maior a densidade do ar e a força de arrasto, fazendo com que mais combustível seja necessário para manter a altitude nominal. Esse é um exemplo de trade-off que tornam problemas de otimização complexos.

As variáveis do problema são definidas por três números inteiros I, D e Q que representam o número de órbitas (revoluções) por dia:

$$rev = I + Q / D. \tag{7.1}$$

Nesse caso, após precisamente D dias, o satélite irá completar uma quantidade inteira de revoluções, repetindo seu traçado. É importante observar que os parâmetros I e Q podem ser utilizados para selecionar a altitude desejada de projeto, visto que quanto mais baixa a altitude orbital, maior o número de revoluções por dia.

Após a seleção da órbita, a carga útil pode ser projetada. Dada a órbita e a resolução de imagem necessária, há muitas maneiras de obter uma estimativa para a massa da carga útil e para o consumo de energia. Neste trabalho, foram utilizadas as equações paramétricas apresentadas em Wertz et al. (2011).

A partir dos parâmetros estimados da câmera e da órbita, a massa da carga útil (W_d) pôde ser obtida e utilizada em uma relação paramétrica de Wertz et al. (2011) para estimar a massa seca do satélite (m_d), dada por:

$$m_d = W_p / 0.31. \quad (7.2)$$

Posteriormente, esta massa seca foi utilizada na estimativa da área de seção transversal, que é utilizada no cálculo da massa de propelente (m_p). Para este trabalho, foi utilizada a abordagem descrita em Chagas et al. (2015) para relacionar a massa seca do satélite (m_d) com a área da seção transversal (A_d). Essa relação foi obtida através da análise dos dados de missões espaciais anteriores, obtidos no banco de dados Union of Concerned Scientists (UCS). Com isso, obteve-se:

$$A_d = 0{,}021647 \times m_d. \quad (7.3)$$

Outro parâmetro importante a ser computado para calcular a massa do propelente é o tempo de reentrada na atmosfera terrestre. As grandes agências espaciais têm seguido orientações a fim de reduzir os debris na órbita terrestre, e uma destas orientações é que um satélite com órbita LEO deve reentrar na atmosfera dentro de 25 anos após o seu fim de vida (ANSELMO; PARDINI, 2015). Para isso, o satélite pode realizar uma manobra orbital após a sua vida útil para diminuir o seu perigeu até um ponto onde o arrasto atmosférico seja suficientemente alto para fazer o satélite reentrar. Porém, esta manobra pode exigir uma quantidade significativa de propelente dependendo da órbita.

Este problema conceitual utilizou a abordagem de Barroca (2019). Em seu trabalho, foi utilizado um pacote de avaliação de detritos (NASA DAS v2.0.2) para construir uma função que, dado o coeficiente balístico do satélite S_{bc} e o apogeu da órbita nominal h_a, fornece o perigeu h_p necessário para que a reentrada ocorra após 25 anos do fim da vida útil. 360 pontos aleatórios foram gerados para o apogeu h_a e para o coeficiente balístico S_{bc} entre 500 e 800 km e entre 0,0001 e 0,9 kg/m², respectivamente. Além disso, foi considerado o pior cenário em termos de fluxo solar e também a inclinação da órbita foi definida como 98º, que é o valor aproximado de todas as órbitas heliossíncronas (SSO). A equação obtida foi:

$$h_p = -2{,}96741 \ln(S_{bc}) \times [38{,}8249 \exp(-0{,}001 h_a) - 50{,}7627] + 1017{,}1592. \quad (7.4)$$

Para uma estimativa altamente precisa desta equação paramétrica, muitas variáveis devem ser consideradas. Porém, nas fases iniciais da concepção conceitual da missão este tipo de precisão não é necessário, uma vez que diversos parâmetros relacionados ao satélite ainda não são conhecidos.

A massa do satélite (S_m) foi estimada utilizando uma relação paramétrica de primeira iteração de Wertz et al. (2011) dada por:

$$S_m = 1{,}27 \times m_d. \tag{7.5}$$

Finalmente, a massa do propelente (mp) foi calculada utilizando a equação mostrada em Wertz et al. (2011) que utiliza a massa total do satélite estimada (Sm) e a variação de velocidade total da missão (ΔVt), dada por:

$$m_p = S_m \left(1 - e^{-\frac{\Delta V}{g0 \cdot Isp}}\right), \tag{7.6}$$

onde g_0 é a aceleração da gravidade na superfície terrestre (9,80665 m/s) e Isp é o impulso específico do propelente, que é 225s para hidrazina (WERTZ et al. 2011). A Tabela 7.1 apresenta os valores dos parâmetros configurados para a simulação.

Tabela 7.1. Configurações iniciais da simulação.

Parâmetro	Valor
I	Intervalo de variação: [13, 15]
D	Intervalo de variação: [1, 60]
Q	Intervalo de variação: [1, D-1]
I_{sp}	225 s
Área total (A_t) porcentagem da área de seção transversal (A_{cs}) [1]	180%
Coeficiente de arrasto (C_d)	2,2
Excentridade da órbita nominal (e)	0,0
Ciclo de vida da missão	4 anos

(continua)

Tabela 7.1. Conclusão.

Parâmetro	Valor
Modelo de densidade atmosférica [2]	1 ano com F10.7 = 225 e 3 anos com F10.7 = 175
Resolução da missão (Res_N)	20 m
Erro de lançamento no semi-eixo maior	20.000 km
Erro de lançamento na inclinação	0,015°

1 Esta é a área total do satélite em relação à área da seção transversal do satélite. Neste caso, a área total é 1,8 vezes a área da seção transversal.

2 A densidade atmosférica média foi obtida a partir do modelo NRLMSISE-00.

Fonte: Barroca (2019).

Portanto, o modelo de otimização do problema juntamente com as restrições é dado por:

$$\text{Minimizar:} \quad S_m\,(I, D, Q), \tag{7.7}$$

$$\text{Onde:} \quad S_m = m_d + m_p, \tag{7.8}$$

$$\text{Restrição 1:} \quad FOV_{payload} - 1.05 \times FOV_{min} \geq 0, \tag{7.9}$$

$$\text{Restrição 2:} \quad 13 \leq I \leq 15, \tag{7.10}$$

$$\text{Restrição 3:} \quad 1 \leq D \leq 60, \tag{7.11}$$

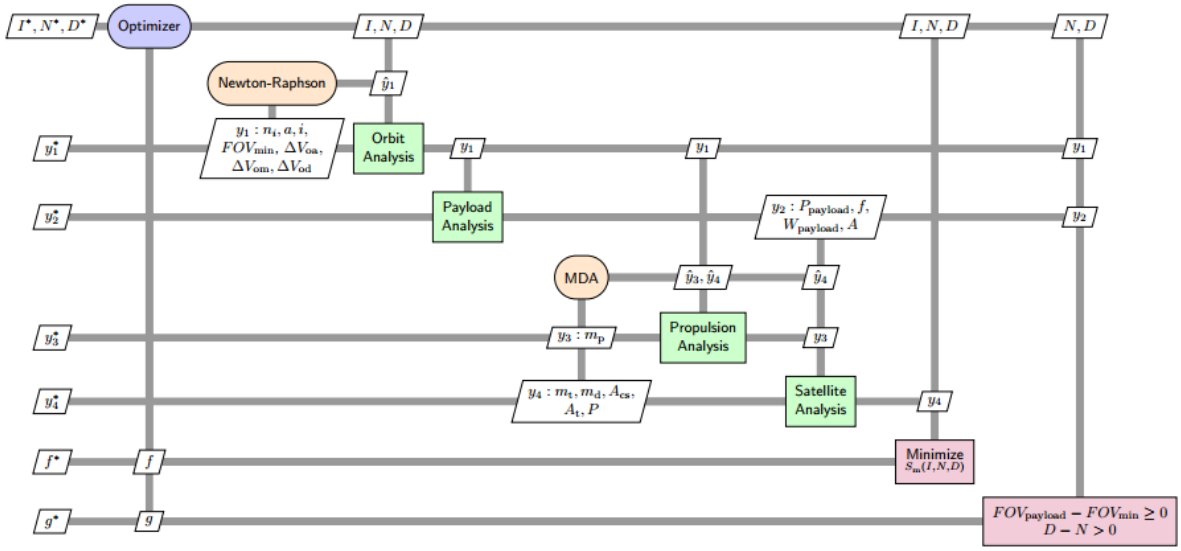
$$\text{Restrição 4:} \quad 1 \leq Q < D, \tag{7.12}$$

onde $FOV_{payload}$ é o campo de visão da carga útil, FOV_{min} é o campo de visão mínimo para que seja possível obter imagens de toda a Terra na órbita nominal.

Após a definição do problema, a arquitetura MDO foi definida, integrando o modelo de otimização, variáveis de design e otimizadores. A arquitetura adotada foi a arquitetura de métodos multidisciplinares (MDF - Multidisciplinary Feasible) devido às suas vantagens de sempre retornar um projeto de sistema que respeite as restrições do problema de otimização e deixe o otimizador ter sob controle direto apenas as variáveis de projeto, função objetivo e restrições

(MARTINS; LAMBE, 2013). A arquitetura é apresentada no formato de matriz de estrutura de projeto estendida (XDSM - eXtended Design Structure Matrix) na Figura 7.1.

Figura 7.1. Diagrama XDSM para o design conceitual da espaçonave.



Fonte: Barroca (2019).

É importante pontuar que este modelo foi simplificado e que simulações com maior fidelidade exigiriam modelos com alto custo computacional, o que tornaria a busca durante uma fase de projeto de conceito não conveniente do ponto de vista prático. No entanto, este problema é suficiente para medir o desempenho dos algoritmos propostos neste trabalho e como eles podem ser utilizados para reduzir o tempo necessário para obter a solução ótima.

Uma prévia busca extensiva foi realizada no espaço de projeto ao variar os valores para cada variável e encontrar a combinação que resultasse no melhor valor da função objetivo, que representa a menor massa do satélite. Para isso, como os valores de I variam entre 13 e 15, os valores de D variam de 1 a 60 e os valores de Q variam de 1 a D-1, todas as 10620 combinações possíveis foram executadas. Essas 10620 avaliações da função objetivo (NFE) são uma quantidade importante, visto que qualquer algoritmo que levar menos avaliações para atingir o ótimo passará a ser mais vantajoso do que a busca

extensiva. Por outro lado, um algoritmo que levar mais que 10620 NFEs para atingir o ótimo não vale a pena ser utilizado, visto que seria mais interessante utilizar a busca extensiva manual do que uma meta-heurística.

A solução ótima do problema é apresentada na Tabela 7.2 e a experimentação numérica realizada foi apresentada na Subseção 4.3.

Tabela 7.2. Solução ótima.

Variáveis	Valores
I	14
D	59
Q	60
Semi-eixo maior da órbita nominal (a)	6944,284 km
Inclinação da órbita nominal (i)	97,656°
FOV_{min}	4,469°
ΔV para aquisição de órbita (ΔV_{oa})	30,768 m/s
ΔV para manutenção de órbita (ΔV_{om})	217,983 m/s
ΔV para manobra de desorbitação (ΔV_{od})	0 m/s
Massa seca (m_d)	175,952 kg
Área de seção transversal (A_{cs})	3,801 m²
Área total (At)	6,841 m²
Massa de propelente (m_p)	20,997 kg
Massa total (m_t)	196,949 kg
Potência (P)	114,769 W

Fonte: Barroca (2019).

Os algoritmos A-GEO2var_5 e A-GEO2real2_AA3 foram executados para esse problema e estes foram comparados com o A-GEO2. Foram realizadas 100 execuções independentes para cada algoritmo e foi obtida a média aritmética dos NFEs obtidos ao final de cada execução e a quantidade de execuções onde o ótimo global foi atingido. Estes resultados são apresentados Tabela 7.3.

Tabela 7.3. Resultados da simulação.

Algoritmo	Número médio de avaliações da função objetivo	Quantidade de execuções em que atingiu o ótimo
Busca extensiva	10.620	Não aplicável
A-GEO2	6.922	100
A-GEO2var_5	40.132	21
A-GEO2real2_AA3	99.278	2

Fonte: Produção do autor.

Pela Tabela 7.3, observa-se que o algoritmo que obteve o melhor desempenho foi claramente o A-GEO2. Este algoritmo precisou, na média, de aproximadamente 7 mil avaliações da função objetivo para atingir o ótimo. Os demais algoritmos levaram mais que 5 vezes a quantidade de avaliações para encontrar o ótimo global. O A-GEO2var_5 obteve uma média de cerca de 40 mil NFEs, ou seja, precisou realizar cerca de 4 vezes mais avaliações da função do que busca extensiva. Porém, este algoritmo ainda foi superior em relação ao A-GEO2real2_AA3, que atingiu o ótimo em apenas 2 das 100 execuções.

No caso do A-GEO2real2_AA3, um dos possíveis motivos para seu baixo desempenho é que os valores das variáveis D e Q no ótimo global são 60 e 59, respectivamente, ou seja, eles estão exatamente na borda do espaço viável de projeto. Diferentemente da codificação binária onde uma solução nunca está fora dos limites superiores e inferiores das variáveis, os algoritmos com codificação real podem encontrar soluções fora do espaço viável, porém essas soluções são penalizadas com a utilização da penalidade externa. A

penalização aplicada é proporcional ao quanto a variável está distante da parte viável do espaço de projeto. Ou seja, por exemplo, quanto maior é o valor da variável D, acima de 60, maior é a penalidade e menos chance a solução tem no momento em que elas são ranqueadas.

Um outro possível motivo para o baixo desempenho do A-GEO2real2_AA3 é que o problema usado como exemplo tem variáveis inteiras. Isso significa que a implementação dos algoritmos teve de levar em conta a conversão dos fenótipos das variáveis de projeto para um número inteiro. O A-GEO2real2_AA3 codifica suas variáveis com números reais, então em cada mutação realizada em uma variável a cada iteração, seu valor era truncado, mudando por exemplo de 13,26 para 13. O truncamento também precisou ser realizado nas versões binárias, pois os bits de uma variável são mapeados para números reais. Por exemplo, a variável I possui o intervalo de variação [13, 15] e 2 bits a codificam. Esses 2 bits resultam em 4 possíveis combinações, que são os valores 0, 1, 2 e 3. Quando mapeados para o intervalo de variação, essas combinações resultam nos valores 13, 13,66, 14,33 e 15. Portanto, foi necessário truncar estes valores, o que não ocorre se o problema possui variáveis contínuas.

Nota-se que o A-GEO2 obteve um desempenho melhor que o A-GEO2var_5. Isso significa que mutar um bit da população inteira, como é feito no A-GEO2, leva a melhores resultados do que mutar um bit para cada variável de projeto, como é feito no A-GEO2var. Ao ordenar os bits com base no valor da função objetivo, o A-GEO2 dá mais chances para mutar o bit que resulta na melhor solução dentre as geradas, diferentemente do A-GEO2var_5, onde o ranqueamento é realizado para cada variável de projeto e o valor da função objetivo deve ser calculado novamente após as mutações. Então mesmo que o bit de cada variável que resulte no melhor valor da função seja o escolhido para mutar no A-GEO2var_5, não necessariamente a nova solução gerada após mutar todos os bits será superior à melhor solução encontrada até o momento. Aqui cabe pontuar que nos trabalhos de De Sousa (2002) e De Sousa et. al. (2003), foi observado que a abordagem de modificar apenas um bit por iteração, ao invés de um bit por variável, parece ser mais vantajosa quando o espaço de

projeto apresenta outros tipos de restrição, além das laterais. No problema do projeto conceitual de sistema espacial tratado aqui, há uma restrição desigualdade ($Q \leq D$), que é inclusive dinâmica, já que D é uma das variáveis de projeto. Esta característica do problema, uma restrição desigualdade dinâmica, é o que provavelmente levou ao melhor desempenho do A-GEO2 sobre o A-GEO2var_5.

Os resultados observados com a aplicação de diferentes versões do GEO adaptativo ao problema de projeto conceitual de um sistema espacial ressalta a importância da escolha adequada do algoritmo de otimização em função das características do problema abordado.

8 CONCLUSÕES

O A-GEO, algoritmo que foi o ponto de partida deste trabalho, é um algoritmo que tornou adaptativo o único parâmetro livre que o GEO continha, não sendo mais necessária a etapa de ajuste de parâmetros. Porém, algumas lacunas como a maneira de ordenar a população de bits e a adaptação dele para operar diretamente com variáveis contínuas ainda estavam abertas. Essas lacunas foram estudadas e, dentre os vários algoritmos de otimização implementados neste trabalho, dois deles foram os que atingiram o melhor desempenho. Estes algoritmos foram denominados A-GEO2var_5 e A-GEO2real2_AA3 e possuem codificação binária e real, respecitvamente. Ambos algoritmos possuem seus parâmetros livres adaptados, sendo o AGEO2var_5 com uma abordagem adaptativa e o A-GEO2real2_AA3 com uma abordagem semi auto-adaptativa.

O ganho de desempenho obtido pelas novas versões quando comparado ao A-GEO2 foi nítido para as funções teste, exceto para a função Rosenbrock que foi a única função onde o A-GEO foi melhor que o A-GEO2var_5. Nas demais funções, exceto na Griewangk, a codificação real foi muito superior em relação a codificação binária. Conforme foi observado no algoritmo A-GEO2var_5, conclui-se que o ranqueamento de bits por variável de projeto atinge resultados superiores quando comparado ao ranquamento de bits da população inteira utilizado no A-GEO2, em problemas sem restrições ou apenas restrições laterais.

Do ponto de vista da mutação porcentagem utilizada nos algoritmos com codificação real, esta atingiu desempenhos superiores às outras duas maneiras de alterar as variáveis de projeto testadas. Isso mostrou que limitar o valor de σ a partir de uma porcentagem do intervalo de variação da variável foi mais vantajoso do que realizar aleatóriamente grandes e pequenas mutações nas variáveis, principalmente porque diferentes problemas possuem diferentes tamanhos de espaço de projeto. Além disso, o fato das versões GEOreal2 realizarem diversas mutações em cada variável de projeto, assim como o GEOvar, foi mais vantajoso do que alterar apenas uma variável por vez.

Grandes e pequenas alterações de uma variável na mesma iteração permitiram que houvesse uma melhor exploração do espaço de projeto a partir de uma única solução, obtendo novas soluções próximas e distantes da solução atual.

Aplicar e verificar as duas melhores versões desenvolvidas com um conjunto maior de funções teste do CEC 2017 permitiu avaliar com mais robustez o desempenho dos mesmos com diversas funções com parâmetros reais. Tanto ao utilizar dimensão 10 quanto 30, observou que tanto o A-GEO2var_5 quanto o A-GEO2real2_AA3 superaram o A-GEO2. Ao utilizar dimensão 10, o algoritmo com codificação real foi superior, enquanto que a versão real foi a que obteve o melhor desempenho utilizando dimensão 30. Porém, estes atingiram um desempenho pior quando comparados a um dos algoritmos que competiu na competição, mostrando que as duas principais implementações geradas neste trabalho ainda não possuem capacidade de competir diretamente com algoritmos de alto desempenho como o EBOwithCMAR.

A aplicação do A-GEO2var_5 e do A-GEO2real2_AA3 no problema conceitual de sistema espacial apresentou piores resultados do que a busca extensiva, mantendo o A-GEO2 como o único destes algoritmos que atinge o ótimo global em menor quantidade de avaliações da função objetivo do que a busca extensiva. Observou-se que os dois algoritmos desenvolvidos podem não ser eficientes quando utilizados em um problema com variáveis discretas. Além disso, o fato do problema utilizado possuir uma restrição dinâmica entre as variáveis de projeto, somado ao ótimo estar próximo do limite das variáveis, torna mais difícil a busca no espaço de projeto para os algoritmos de otimização. Quando o espaço de projeto apresenta outros tipos de restrição, além das laterais, a abordagem de modificar apenas um bit por iteração, ao invés de um bit por variável, parece ser mais vantajosa. Isso foi o que provavelmente levou ao melhor desempenho do A-GEO2 em relação aos algoritmos desenvolvidos quando aplicados no problema conceitual de um sistema espacial.

Ainda há muita pesquisa a ser desenvolvida visando melhorar o mecanismo do GEO adaptativo. Dentre trabalhos futuros, estão:

- Utilização de mutação correlacionada em versões com codificação real;
- hibridização com outros algoritmos para a realização de busca local mais eficiente;
- tornar o algoritmo com codificação real totalmente auto-adaptativo;
- investigar novas maneiras de adaptar o parâmetro τ;
- aprofundar o entendimento da estagnação do A-GEO2var;
- compreender a interdependência entre os parâmetros τ e ρ, visando obter melhores estratégias para variá-los sinergicamente na procura do ótimo.

Embora a utilização de uma abordagem adaptativa para o GEO venha mostrando-se vantajosa, como apresentado em Barroca (2019) e no presente estudo, ela ainda está em uma fase embrionária de desenvolvimento. O presente trabalho de mestrado é mais um passo na introdução do controle adaptativo de parâmetros no GEO, de forma a torná-lo cada vez mais eficiente e amigável ao usuário em problemas de otimização.

REFERÊNCIAS BIBLIOGRÁFICAS

ALETI, A.; MOSER, I. A systematic literature review of adaptive parameter control methods for evolutionary algorithms. **ACM Computing Surveys (CSUR)**, v. 49, n. 3, p. 1-35, 2016.

ALI, M. M.; CHAROENCHAI, K.; ZABINSKI, Z. B.; A Numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. **Journal of Global Optimization,** v.31, p. 635-672, 2005.

ANSELMO, L.; PARDINI, C.; Compliance of the Italian satellites in low Earth orbit with the end-of-life disposal guidelines for Space Debris Mitigation and ranking of their long-term criticality for the environment. **Acta Astronautica**, v. 114, p. 93-100, 2015.

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.: s.n.], 2017.

BARROCA, E. **A new adaptive evolutionary algorithm for design optimization**. 2019. Tese (Doutorado em Engenharia e Gerenciamento de Sistemas Espaciais) - Instituto Nacional de Pesquisas Espaciais (INPE), São Jose dos Campos, SP, 2019.

BOETTCHER, S.; PERCUS, A. G. Optimization with extremal dynamics. **Complexity**, v. 8, n. 2, p. 57-62, 2002.

BOUDJEMAI, A.; BOUANANE, M. H.; MERAD, L.; MOHAMMED, A. S. Small satellite structural optimization using genetic algorithm approach. In: INTERNATIONAL CONFERENCE ON RECENT ADVANCES IN SPACE TECHNOLOGIES, 3., 2007. **Proceedings…** IEEE, 2007. p. 398-406.

BOUSSAID, I.; LEPAGNOT, J; SIARRY, P.; A survey on optimization metaheuristics. **Information Sciences,** v. 237, p. 82-117, 2013.

CHAGAS, R. A. J.; DE ALBUQUERQUE, B. F. C.; LOPES, R. A. M.; DE SOUSA, F. L. Towards the automation of concurrent space systems conceptual design through multidisciplinary design optimization. In: ABCM INTERNATIONAL CONGRESS OF MECHANICAL ENGINEERING, 23., 2015. **Proceedings…** 2015.

ČREPINŠEK, M.; LIU, S.; MERNIK, M. Exploration and exploitation in evolutionary algorithms: A survey. **ACM Computing Surveys (CSUR)**, v. 45, n. 3, p. 1-33, 2013.

CUCO, A. P. C.; DE SOUSA, F. L.; SILVA NETO, A. J. A multi-objective methodology for spacecraft equipment layouts**. Optimization and Engineering**, v. 16, n. 1, p. 165-181, 2015.

DA LUZ, L. B.; DE SOUSA, F. L.; CHAGAS, R. A. J. Implementação adaptativa de variante do algoritmo de Otimização Extrema Generalizada (GEO). In: SEMINÁRIO INTEGRADO DE SOFTWARE E HARDWARE, 48.,2021. **Anais...** SBC, 2021. p. 271-278.

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

DE FALCO, I.; DELLA C. A.; MAISTO, D. S. U.; TARANTINO, E. Biological invasion–inspired migration in distributed evolutionary algorithms. **Information Sciences**, v. 207, p. 50-65, 2012.

DE SOUSA, F. L. **Otimização extrema generalizada: um novo algoritmo estocástico para o projeto ótimo**. 2002. Tese (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2002.

DE SOUSA, F. L.; RAMOS, F. M.; PAGLIONE, P.; GIRARDI, R. M. New stochastic algorithm for design optimization. **AIAA Journal**, v. 41, n. 9, p. 1808-1818, 2003.

DE SOUSA, F. L.; SOEIRO, F. J. C. P.; SILVA NETO, A. J.; RAMOS, F. M. Application of the generalized extremal optimization algorithm to an inverse radiative transfer problem. **Inverse Problems in Science and Engineering**, v. 15, n. 7, p. 699-714, 2007.

EIBEN, Á. E.; HINTERDING, R.; MICHALEWICZ, Z. Parameter control in evolutionary algorithms. **IEEE Transactions on Evolutionary Computation**, v. 3, n. 2, p. 124-141, 1999.

EIBEN, A. E., SMITH, J.; Parameters and parameter tuning. In: EIBEN, A. E., SMITH, J. **Introduction to evolutionary computing**. 2.ed. [S.l.]: Springer, 2015a.

EIBEN, A. E.; SMITH, J. From evolutionary computation to the evolution of things. **Nature**, v. 521, n. 7553, p. 476-482, 2015.

FISTER, I.; BREST, J. I. A.; GALVEZ, A.; DEB, S. On selection of a benchmark by determining the Algorithms' qualities. **IEEE Access**, v. 9, p. 51166-51178, 2021.

FREITAS, V. L. S.; DE SOUSA, F. L.; MACAU, E. E. N. Reactive model for autonomous vehicles formation following a mobile reference. **Applied Mathematical Modelling**, v. 61, p. 167-180, 2018.

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**. 2006. Tese (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2006.

GALSKI, R. L.; PATIRE JUNIOR, H.; DE SOUSA, F. L.; HINCKEL, J. N.; 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…** 2011. p. 407-414.

GENTILE, L.; FILIPPI, G.; MINISCI, E.; BARTZ-BEIELSTEIN, T.; VASILE, M. Preliminary spacecraft design by means of structured-chromosome genetic algorithms. In: IEEE CONGRESS ON EVOLUTIONARY COMPUTATION (CEC), 2020. **Proceedings…** IEEE, 2020. p. 1-8.

GREINER, D.; PERIAUX, J.; QUAGLIARELLA, D.; MAGALHAES-MENDES, J.; GALVÁN, B. Evolutionary algorithms and metaheuristics: applications in engineering design and optimization. **Mathematical Problems in Engineering**, v. 2018, 2018.

HARE, W.; NUTINI, J.; TESFAMARIAM, S. A survey of non-gradient optimization methods in structural engineering. **Advances in Engineering Software**, v. 59, p. 19-28, 2013.

HINCKLEY J. R. D.; ENGLANDER, J.; HITT, D. Multi-objective optimization of spacecraft trajectories for small-body coverage missions. In: AAS/AIAA SPACE FLIGHT MECHANICS MEETING, 2017. **Proceedings…**2017.

ILHAN, I. Mobile device-based test tool for optimization algorithms. **Computer Applications in Engineering Education**, v. 24, n. 5, p. 744-754, 2016.

JAMIL, M.; YANG X. S.; A literature survey of benchmark functions for global optimisation problems. **International Journal of Mathematical Modelling and Numerical Optimisation**, v. 4, no. 2, p. 150-194, 2013.

KALITA, H.; THANGAVELAUTHAM, J. Automated design of CubeSats using evolutionary algorithm for trade space selection. **Aerospace**, v. 7, n. 10, p. 142, 2020.

KARAFOTIAS, G.; HOOGENDOORN, M.; EIBEN, Á. E. Parameter control in evolutionary algorithms: Trends and challenges. **IEEE Transactions on Evolutionary Computation**, v. 19, n. 2, p. 167-187, 2014.

KUMAR, A.; MISRA, R.; SINGH, D. Improving the local search capability of effective butterfly optimizer using covariance matrix adapted retreat phase. In: IEEE CONGRESS ON EVOLUTIONARY COMPUTATION (CEC), 2017. **Proceedings…** IEEE, 2017. p. 1835-1842.

LAU, V.; SOUSA, F. L. D.; GALSKI, R. L.; ROCCO, E. M.; BECCENERI, J. C.; SANTOS, W. A. D.; SANDRI, S. A. A multidisciplinary design optimization tool for spacecraft equipment layout conception. **Journal of Aerospace Technology and Management**, v. 6, p. 431-446, 2014.

LIANG, J. J.; QU, B. Y.; SUGANTHAN, P. N.; HERNÁNDEZ-DÍAZ, A. **Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization**. Singapore: Zhengzhou University; China and Nanyang Technological University, 2013a.

LIANG, J. J.; QU, B. Y.; SUGANTHAN, P. N. **Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization**. Singapore: Zhengzhou University; Nanyang Technological University, 2013b.

MAIER, H. R.; KAPELAN, Z.; KASPRZYK, J.; KOLLAT, J.; MATOTT, L. S.; CUNHA, M. C.; DANDY, G. C.; GIBBS, M. S.; KEEDWELL, E.; MARCHI, A.; OSTFELD, A.; SAVIC, D.; SOLOMATINE, D. P.; VRUGT, J. A.; ZECCHIN, A. C.; MINSKER, B. S.; BARBOUR, E. J.; KUCZERA, G.; PASHA, F.; CASTELLETTI, A.; GIULIANI, M.; REED, P. M. Evolutionary algorithms and other metaheuristics in water resources: Current status, research challenges and future directions. **Environmental Modelling & Software**, v. 62, p. 271-299, 2014

MAIER, H. R.; RAZAVI, S.; KAPELAN, Z..; MATOTT, L. S.; KASPRZYK, J.; TOLSON, B. A.; Introductory overview: optimization using evolutionary algorithms and other metaheuristics. **Environmental Modelling & Software**, v. 114, p. 195-213, 2019.

MAINENTI-LOPES, I.; SOUZA, L. C. G.; DE SOUSA, F. L. 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.

MAINENTI-LOPES, I.; SOUZA, L. C. G.; DE SOUSA, F. L. 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.

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.

MEZIANE-TANI, I.; MÉTRIS, G.; LION, G.; DESCHAMPS, A.; BENDIMERAD, F. T.; BEKHTI, M. Optimization of small satellite constellation design for continuous mutual regional coverage with multi-objective genetic algorithm. **International Journal of Computational Intelligence Systems**, v. 9, n. 4, p. 627-637, 2016.

MOLGA, M.; SMUTNICKI, C. **Test functions for optimization needs**. 2005. Disponível em: http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf.

MUELLER, C. T.; OCHSENDORF, J. A. Combining structural performance and designer preferences in evolutionary design space exploration. **Automation in Construction**, v. 52, p. 70-82, 2015.

MURAOKA, I.; GALSKI, R. L.; DE SOUSA, F. L.; 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.

PAEK, S. W.; KIM, S.; DE WECK, O. Optimization of reconfigurable satellite constellations using simulated annealing and genetic algorithm. **Sensors**, v. 19, n. 4, p. 765, 2019.

SLOWIK, A.; KWASNICKA, H.; Evolutionary algorithms and their applications to engineering problems. **Neural Computing and Applications**, v. 32, n. 16, p. 12363-12379, 2020.

SRINIVASAN, D.; SEOW, T. H. Particle swarm inspired evolutionary algorithm (PS-EA) for multiobjective optimization problems. In: THE CONGRESS ON EVOLUTIONARY COMPUTATION, 2003. **Proceedings…** IEEE, 2003. p. 2292-2297.

SUGANTHAN, P. N.; HANSEN, N.; LIANG, J. J.; DEB, K.; CHEN, Y. P.; AUGER, A.; TIWARI, S. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. **KanGAL Report**, v. 2005005, 2005.

SURJANOVIC, S.; BINGHAM, D**. Virtual library of simulation experiments**: test functions and datasets. Simon Fraser University, Burnaby, BC, Canada, 2013. Disponível em: https://www.sfu.ca/~ssurjano/optimization.html. Acesso em: jan. 2022.

THEODOSSIOU, N. P.; KOUGIAS, I. P. Harmony search algorithm. **WIT Transactions on State-of-the-art in Science and Engineering**, v. 56, 2012.

VENT, W. **Rechenberg, Ingo, evolutionsstrategie:** optimierung technischer systeme nach prinzipien der biologischen evolution. Stuttgart: Frommann-Holzboog-Verlag, 1973.

WERTZ, J. R.; EVERETT, D. F.; PUSCHELL, J. J. **Space mission engineering**: the new SMAD. [S.l.]: Microcosm Press, 2011.

ZHAO, W.; WANG, L.; ZHANG, Z. A novel atom search optimization for dispersion coefficient estimation in groundwater. **Future Generation Computer Systems**, v. 91, p. 601-610, 2019.