Universidade Federal Fluminense

MÁRCIO ECCARD KORT-KAMP

Análise do Método de Otimização Alcateia em Problemas de Engenharia

SANTO ANTÔNIO DE PÁDUA
2020

MÁRCIO ECCARD KORT-KAMP

Análise do Método de Otimização Alcateia em Problemas de Engenharia

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia da Universidade Federal Fluminense, como requisito parcial para obtenção do título de Mestre em Modelagem Computacional em Ciência e Tecnologia. Área de Concentração: Otimização aplicada a engenharia.

Orientador:

Cleber Almeida Corrêa Junior

Coorientadora:

Rosilene Abreu Portella Corrêa

Universidade Federal Fluminense

SANTO ANTÔNIO DE PÁDUA

2020

Ficha catalográfica automática - SDC/BEM
Gerada com informações fornecidas pelo autor

K85a Kort-kamp, Márcio Eccard
Análise do Método de Otimização Alcateia em Problemas de Engenharia / Márcio Eccard Kort-kamp ; Cleber Almeida Corrêa Junior, orientador ; Rosilene Abreu Portella Corrêa, coorientadora. Volta Redonda, 2020.
101 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Volta Redonda, 2020.

DOI: http://dx.doi.org/10.22409/PPG-MCCT.2020.m.00357378750

1. Método Alcateia. 2. Otimização. 3. Métodos Estocásticos. 4. Produção intelectual. I. Corrêa Junior, Cleber Almeida, orientador. II. Corrêa, Rosilene Abreu Portella, coorientadora. III. Universidade Federal Fluminense. Escola de Engenharia Industrial e Metalúrgica de Volta Redonda. IV. Título.

CDD -

Bibliotecário responsável: Sandra Lopes Coelho - CRB7/3389

Análise do Método de Otimização Alcateia em Problemas de Engenharia

Márcio Eccard Kort-Kamp

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia da Universidade Federal Fluminense, como requisito parcial para obtenção do título de Mestre em Modelagem Computacional em Ciência e Tecnologia. Área de Concentração: Otimização aplicada a engenharia.

Aprovada por:

Prof Cleber de Almeida Corrêa Junior, D.Sc/MCCT-UFF

(Presidente)

Prof. Thiago Jordem Pereira D.Sc/MCCT-UFF

Prof. Luciano dos Santos Rangel D.Sc/FeMASS-Macaé

Santo Antônio de Pádua, 30 de novembro de 2020

Agradecimentos

O meu primeiro agradecimento vai a Deus que por ser o criador de todas as coisas me fez capaz de aprender o que me transmitem como conhecimento, capaz de ver a vida sob novos horizontes e capaz de reconhecer e enfrentar todos os desafios que a carreira impõe.

A minha esposa Adriana por toda palavra de ânimo.

Ao meu filho Pedro pela paciência nas minhas ausências.

Ao meus pais Acelino e Nercy, pelo interesse em me manter ligado aos estudos para obter da vida o que de melhor ela pode me proporcionar.

Aos meus professores e orientadores, Cleber e Rosilene Corrêa, que não se esquivaram de mostrar os pontos chaves para que eu pudesse atender às exigências que um trabalho como este enseja.

Agradeço também aos meus demais professores do programa de mestrado, Thiago Pereira, Ricardo Silveira, Tibério Vale e Gustavo Semaan, por todo ensinamento passado.

Aos colegas de curso, que também tiveram sua participação nesse empreendimento. Em especial, Lucas Kort e Andressa Alves.

Resumo

Frequentemente são observados problemas de otimização em diversas áreas da Ciência e da Engenharia. No entanto, devido à complexidade de algumas funções que modelam estes fenômenos, é preciso que se tenha uma atenção maior aos estudos e desenvolvimento de métodos numéricos que se mostrem eficientes e capazes de reproduzir, realisticamente, soluções para os modelos supracitados. Diante disso, buscou-se apresentar e analisar o método de otimização bio-inspirado Alcateia, sendo seu objetivo analisar o desempenho do referido método aplicado na resolução de problemas de otimização. O algoritmo Alcateia é um método que foi criado e desenvolvido pelos professores D.Sc. Cleber de Almeida Corrêa Júnior e D.Sc. Rosilene Abreu Portella Corrêa a partir da análise/observação do padrão de comportamento de lobos durante sua caça. Como ocorre com grupos de lobos, durante a caça, que procuram pela presa mais vulnerável, a Alcateia, também, está em uma busca constante de troca de líderes, denominado como lobo alfa, o que indica a presa que deve ser abatida e que coordena os demais lobos. Foi a partir destas observações que foi criado o Método Estocástico Alcateia para os problemas de otimização. No contexto do presente estudo, a sua aplicação se dá em problemas de funções multimodais com e sem restrições e, ainda, em equações de Engenharia que são bastante difundidos em literaturas específicas. Realizou-se um estudo comparativo entre os resultados encontrados com o uso da metodologia e os resultados presentes em literaturas que foram obtidos por meio de outros métodos de otimização. Verificou-se que o modelo utilizado encontrou valores ótimos para todas as equações testadas, comprovando, dessa forma, a eficiência e a robustez do método na resolução de problemas de otimização.

Abstract

Optimization problems are often observed in various areas of science and engineering. However, due to the complexity of some functions that model these phenomena, greater attention needs to be given to the study and development of numerical methods that are efficient and capable of realistically reproducing solutions for the aforementioned models. In view of this, we tried to present and analyze the bio-inspired optimization method Alcateia, and its objective to analyze the performance of said method applied in the resolution of optimization problems. The Alcateia algorithm is a method that was created and developed by the teachers D.Sc. Cleber de Almeida Corrêa Júnior and D.Sc. Rosilene Abreu Portella Corrêa from the analysis / observation of the behavioral pattern of wolves during their hunting. As with hunting groups of wolves looking for the most vulnerable prey, the wolf pack is also in a constant search for leader change, called the alpha wolf, which indicates the prey to be slaughtered and coordinates the other wolves. It was from these observations that the Stochastic Method Alcateia was created for the optimization problems. In the context of the present study, its application occurs in problems of multimodal functions with and without restrictions, and also in Engineering equations that are quite widespread in literature specific. A comparative study was carried out between the results found using the methodology and the results found in literature that were obtained through other optimization methods. It was found that the model used found optimal values for all equations tested, thus proving the efficiency and robustness of the method in solving optimization problems.

Palavras-chaves

1. Método Alcateia
2. Otimização
3. Métodos Estocásticos
4. Funções multimodais
5. Funções restritas
6. Funções irrestritas
7. Funções de engenharia
8. Identificação de danos em uma viga

Lista de figuras

Lista de tabelas

Sumário

Capítulo 1

1 Introdução

A busca humana por aperfeiçoamento, nos mais variados processos, é crescente. O homem está sempre buscando alcançar melhores resultados e, com isso, se defronta, constantemente, com a necessidade de encontrar soluções ótimas e, para isso, impõe restrições e formaliza variáveis e objetivos. Dessa forma, a natureza matemática do problema emerge. O que descreve a semelhança entre a realidade diária e o idealismo do objeto é esse processo de modelagem que transcreve situações cotidianas em forma matemática com o intuído de observar, estudar e aperfeiçoar as situações do dia-a-dia (CORRÊA JR.; CORRÊA, 2017). A busca do homem pela perfeição, sua sede de conhecimento e o modo como este está sempre à busca de novos métodos, é o que facilita a vida das pessoas. É o resumo desses desejos e suas essências que fez com que fosse criado um ramo da matemática chamado otimização (CORRÊA JR.; CORRÊA, 2017).

Esta busca pela otimização pode ser utilizada em uma infinidade de situações ligadas às áreas das Ciências e da Engenharia. Dentre elas pode-se citar: a alocação de recursos orçamentários, definição de rotas de veículos, criação de tabelas de campeonatos, planejamentos de produção, uso de recursos físicos disponíveis em redes de internet, problemas com despachos econômicos, problema quase-linear de Brown, problemas de cinemática robótica, problemas de direção automotiva, projetos de vaso de pressão, problemas de viga soldada e outros. Muitos destes problemas do dia-a-dia podem ser modelados matematicamente por funções com ou sem restrições. No entanto, estas soluções demandam a determinação de valores extremos (VIANA, 2017).

Poder-se-ia dizer, portanto, que um problema de otimização é aquele que se utiliza de técnicas de determinação de valores máximos e mínimos no sentido de buscar soluções ótimas (ou a melhor solução possível) dentre todas as soluções que forem encontradas em determinado espaço de busca (IZMAILOV; SOLODOV, 2007).

A otimização, muitas vezes, envolve funções complexas e multidimensionais que demandam soluções difíceis ou impossíveis de serem obtidas analiticamente e,

desse modo, foi necessário desenvolver técnicas numéricas que se mostrassem eficientes e capazes de reproduzir soluções para problemas de otimização específicos (VIANA, 2017). Nesse sentido, muitos estudos têm sido desenvolvidos com o objetivo de aplicar técnicas numéricas que ajudem a encontrar a melhor ordem de aproximação ótima destes problemas (SARAMAGO, 2003; KAVISKY; PRADO, 2008; TELLES; RODRIGUES, 2016).

No sentido de resolver os problemas de otimização, tem sido adotada, ao longo dos tempos, muitas metodologias e, dentre elas, pode-se destacar os métodos heurísticos ou aleatórios de otimização (MARTINS, 2004; ZÖRNIG, 2011). Estes métodos admitem que a busca por possíveis soluções ótimas para os problemas de otimização sejam feitas com base em números aleatórios. O objetivo desta escolha é melhorar a ordem de aproximação da variável de interesse e da estratégia de resolução (VIANA, 2017). Dentre os métodos heurísticos ou estocásticos (ou aleatórios) de otimização propostos na literatura, encontra-se o método Alcateia (CORRÊA JR.; CORRÊA, 2017), objeto do presente estudo, e outros, como o Luus Jaakola (LJ) (LUUS; JAAKOLA, 1973), o Particle Swarm Optimization (PSO) (KENNEDY; EBERHART, 1995) e o Simulated Annealing (SA) (KIRKPATRICK; GELATT, 1983).

Ao longo dos anos, muitos trabalhos (CARRILLO, 2007; LARA HERRERA, 2007; GALEANO, 2007; FERREIRA, 2008; LIMA JR. et al., 2009; SERAPIÃO, 2009; SILVA, 2009; CORRÊA JR.; CORRÊA, 2017; CORRÊA, 2013), têm comprovado que estas metodologias tem sido utilizadas com sucesso.

O método Alcateia, que é objeto do presente estudo, tem como base no comportamento de uma alcateia durante seu processo de busca por alguma presa. De modo geral, as alcateias são formadas por grupos de 02 a 15 lobos (o número de lobos depende do tamanho do território, da quantidade de comida e de fatores sociais), no entanto, alguns estudos demonstram que este número pode ser até maior e chegam a citar alcateias de até 36 lobos (BOITANI; CIUCCI, 1995; SMITH; PETERSON; HOUSTON, 2003). Nas alcateias, há uma hierarquia bem definida e o que define essa hierarquia é a postura do lobo, suas características comportamentais e as lutas pela liderança do grupo. Na ordem hierárquica das alcateias, há um casal de lobos dominantes, denominado alfa, que são os primeiros que se alimentam e os únicos que reproduzem dentro da alcateia. Logo a seguir, na ordem hierárquica, vêm os lobos betas e ômegas (CORRÊA JR.; CORRÊA, 2017; CORRÊA, 2013).

O lobo alfa é quem escolhe, durante a caça, qual presa deverá ser abatida naquele momento e, para essa escolha, o alfa busca o animal mais velho, o mais novo ou aquele que está adoentado, ou seja, a mais acessível para ser atacada. Verifica-se, com isso, que o lobo alfa busca a presa ótima (solução ótima para o problema de otimização). Nas ocasiões em que outro lobo quer tomar para si a posição de alfa dentro da alcateia ocorre uma disputa violenta, caracterizada por luta corporal, passando a ser o lobo alfa aquele que vencer a disputa (CORRÊA JR.; CORRÊA, 2017). São estas características das alcateias que formam a base para a modelagem matemática e computacional que dão origem ao método.

Frente às prerrogativas apresentadas, o presente estudo teve por objetivo geral: compreender o conceito de algoritmos de otimização estocástica, buscando verificar e validar o método de otimização bio-inspirado Alcateia. Teve, ainda, como objetivos específicos: testar o funcionamento do algoritmo para programação multimodal sem restrições; testar o funcionamento do algoritmo para programação multimodal com restrições; testar o funcionamento do algoritmo em projetos de engenharia; e, por fim, testar o funcionamento do algoritmo na identificação de danos em uma viga.

De modo a atender aos objetivos do estudo, optou-se por dividi-lo em dois momentos, sendo o primeiro a revisão de literatura e o segundo a apresentação dos resultados e discussão com a literatura analisada. Na revisão de literatura fez-se uma análise sobre aspectos conceituais dos problemas de otimização determinísticos e estocásticos e sobre os métodos estocásticos mais utilizados. E, por fim, uma análise do método estocástico Alcateia, objeto do presente estudo. Após a análise realizada, foram apresentados os resultados do funcionamento do algoritmo em funções multimodais com e sem restrições, em projetos de engenharia e na identificação de danos em uma viga.

Capítulo 2

2 Métodos de otimização

Existem métodos de otimização determinísticos e estocásticos e o que determina qual método de otimização deve ser utilizado, e pode se apresentar mais vantajoso, é a função que se irá trabalhar. De acordo com Silva (2009) ocorrem casos em que os métodos híbridos, com uso de métodos estocásticos, se mostram mais viáveis e, em determinado ponto, há a necessidade de modificar a busca com métodos determinísticos.

No entanto, segundo Brandão (2010), o uso de métodos determinísticos pode se apresentar desvantajosos quando a função é altamente diferençável. Nesses casos, dependendo da função objetivo, os métodos convergem para um ponto estacionário o que faz com que os métodos determinísticos fiquem “presos” (grifo do autor) nos locais ótimos que termina não trazendo um resultado satisfatório para o trabalho desenvolvido.

Nesse sentido, segundo Lima Jr. et al. (2009), começaram a surgir, a partir da segunda metade do século XX, técnicas de otimização com base em processos aleatórios, guiados por heurística, conhecidas como técnicas estocásticas (os métodos estocásticos podem também ser chamados de métodos heurísticos). Estas técnicas começaram a ser desenvolvidas e, à medida que começavam a obter êxito na solução de problemas que até então não se tinha encontrado soluções, começaram a se destacar.

Dentre outros motivos, utilizam-se os métodos estocásticos quando a função objetivo que deve ser minimizada é de difícil representação não-diferenciável, descontínua, não-linear, multimodal. Os métodos estocásticos, segundo Oliveira e Saramago (2005), são os que procuram o ótimo por meio de regras de probabilidade, operando, de maneira aleatória, mas orientada.

Alguns exemplos de métodos estocásticos são o Busca Tatu, o Simulated Annealing, os métodos baseados em população (algoritmos genéricos, evolução diferencial, otimização por colônias de formigas ou por bando de pássaros e outros).

Dentre os métodos estocásticos citados, Silva (2009), destaca os de algoritmos naturais baseados em populações, pelo fato destes se caracterizarem por tentar simular os procedimentos da natureza com o intuito de encontrar soluções para problemas complexos e um possível ótimo global da função objetivo.

Levando-se em conta estes esclarecimentos primários, analisar-se-á a partir deste momento da pesquisa alguns pontos, encontrados na literatura, sobre conceitos de algoritmos de otimização estocástica, e outros conceitos ligados ao tema e importantes para a pesquisa, antes de verificar e validar o método de otimização bio-inspirado Alcateia. A análise dessas literaturas será de suma importância para a discussão que será promovida após a apresentação dos resultados encontrados com o presente estudo.

2.1 Otimização

A otimização é uma realidade que pode ser aplicada nas mais variadas áreas de conhecimento, e até mesmo os animais, de forma instintiva, aplicam a otimização em seu dia-a-dia. Pode-se utilizar como exemplo as abelhas, pois estas constroem seus alvéolos buscando, sempre, uma maneira mais econômica, uma maneira que traga maior volume e uma menor porção de material utilizado. Tendo em vista que os alvéolos não podem ser cilíndricos, por exemplo, a solução encontrada pelas abelhas é que estes tenham a forma de um prisma, pois dessa maneira, as paredes de um alvéolo servem, também, para os alvéolos vizinhos e, dessa forma, elas conseguem guardar um maior volume de mel com uma mínima quantidade de material utilizado (BRANDÃO, 2010).

No que tange, especificamente, à área da Matemática, os problemas de otimização tratam de problemas de maximização ou de minimização de uma função, chamada de função objetivo. Esta função possui uma ou mais variáveis em um determinado espaço de busca ou domínio, sendo que, muitas vezes, ocorrem várias restrições nas variáveis (BASTOS, 2004).

Considera-se problemas de otimização os problemas de maximização ou minimização de função de uma ou mais variáveis num determinado domínio (BRANDÃO, 2010). De modo geral, há um conjunto de restrições em suas variáveis (LIMA JR. et al., 2009).

Os problemas de otimização podem ser, matematicamente, representados, da seguinte forma (HOLTZ, 2005):

Maximizar/Minimizar: $f(x_1, x_2, ..., x_n)$

Satisfazendo:

$$\begin{matrix} g_1(x_1, x_2, ..., x_n) \\ g_2(x_1, x_2, ..., x_n) \\ \vdots \\ g_m(x_1, x_2, ..., x_n) \end{matrix} \quad \begin{matrix} \leq \\ = \\ \geq \end{matrix} \quad \begin{matrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{matrix}$$

em que:

$x_1, x_2, ..., x_n$ – são as variáveis de projeto;

$f(x_1, x_2, ..., x_n)$ – é a função objetivo e

$g_1, g_2, ..., g_m$ – são as restrições.

Segundo Bastos (2004) para que se compreenda melhor os algoritmos de otimização, vale fazer uma análise de alguns conceitos e definições que tem sido utilizado na literatura. Dentre estes conceitos, são listados, abaixo, alguns termos que tem sido, usualmente, utilizados a problemas de otimização. Dentre eles, estão:

- Variáveis de projeto: São aquelas que se alteram durante o processo de otimização. Estas variáveis podem ser contínuas, inteiras ou discretas.
- Restrições: São as funções de igualdade ou desigualdade sobre as variáveis de projeto. Elas condicionam uma ou mais variáveis em uma dada decisão a ser observada na resolução do problema.
- Espaço de Busca: É o conjunto, espaço ou região que compreende as soluções possíveis ou variáveis do projeto do problema a ser otimizado. O espaço de busca é delimitado pelas funções de restrição.
- Função Objetivo: É a função de uma ou mais variáveis do projeto que se deseja otimizar, minimizando-a ou maximizando-a.
- Ponto Ótimo: É o ponto formado pelas variáveis de projeto que extremizam a função objetivo e satisfazem as restrições.
- Valor ótimo: É o valor da função objetivo no ponto ótimo.

Alguns problemas de otimização envolvem, apenas, modelos lineares em que as variáveis são contínuas e apresentam comportamento linear, tanto em relação à função objetivo quanto às restrições. Já outros problemas de otimização envolvem modelos não-lineares que têm como características a exibição de qualquer tipo de não-linearidade ou não continuidade, tanto no que compete à função objetivo quanto a qualquer de suas restrições. De modo geral, estes apresentam a maior dificuldade e complexidade em suas resoluções (GALEANO, 2007) e, nesse sentido, demandam uma análise mais aprofundada.

2.1.1 Métodos Determinísticos

Os métodos de otimização baseados em algoritmos determinísticos são, em sua maioria, clássicos e geram uma sequência determinística de possíveis soluções. Para tanto, é necessário que se determine, ao menos, a primeira derivada da função objetivo em relação às variáveis do projeto. É necessário, que no espaço de busca, a função objetivo seja contínua e diferenciável (BASTOS, 2004).

Segundo Brandão (2010), os métodos determinísticos se mostram mais eficientes quando utilizados em funções contínua, convexas e semi-modais.

Para Oliveira e Saramago (2005), quando se trata de um problema de variáveis discretas, deve-se considerar um espaço de busca com variáveis contínuas que, após a otimização, poderão fornecer uma aproximação das variáveis de projeto para as disponíveis no espaço discreto. No entanto, isso pode gerar um trabalho adicional no que compete à escolha das variáveis discretas mais próximas das contínuas encontradas. Segundo os autores (OLIVEIRA; SARAMAGO, 2005), sempre haverá duas opções de variáveis discretas para cada variável contínua, ou seja, uma imediatamente superior e outra imediatamente inferior.

As técnicas determinísticas, em geral, têm um baixo número de avaliações na função objetivo o que termina fazendo com que tenha convergência rápida e baixo custo computacional (BRANDÃO, 2010).

Os métodos determinísticos apresentam teoremas que lhes garante uma convergência para uma solução ótima que, não necessariamente, apresenta uma solução ótima global. E, tendo em vista, que nos métodos determinísticos a solução encontrada depende do ponto de partida fornecido, pode convergir para um ótimo

local. Desse modo, não apresentam um bom resultado em otimizar funções multimodais, ou seja, funções que possuem vários ótimos locais (BRANDÃO, 2010).

Dentre os métodos determinísticos os mais tradicionais, estão: Máxima Descida, Método de Newton, Método de Quase-Newton, Gradiente Conjugado, Método de Levenberg-Marquadt e Método Simplex (SILVA; SOUZA; SILVA; MARIANI, 2008).

2.1.2 Métodos Estocásticos

Foi somente a partir da segunda metade do século XX que técnicas de otimização baseadas em processos aleatórios, guiadas por alguma heurística (técnicas estocásticas), começaram a ser desenvolvidas e a ganhar destaque. Estas técnicas começaram a ficar evidentes ao obterem algum êxito na solução de problemas que até então se encontravam sem solução (LIMA JR. et al., 2009).

Segundo Brandão (2010), os métodos estocásticos podem ser denominados métodos heurísticos, que é um termo que tem origem na palavra grega *heuriskein* que significa encontrar/descobrir. Estes métodos podem ser utilizados quando a função objetivo a ser minimizada ou maximizada pode ser de difícil representação, não-diferenciável, descontínua, não-linear ou multimodal.

De acordo com Oliveira e Saramago (2005), os métodos estocásticos são aqueles que “[...] buscam o ótimo através de regras de probabilidade operando de maneira aleatória orientada”.

Algumas das vantagens que os algoritmos heurísticos apresentam em relação aos algoritmos determinísticos, segundo Bastos (2004), são:

- Não é necessário que a função objetivo seja continua ou diferenciável.
- Trabalham de modo adequado tanto com parâmetros contínuos quanto com parâmetros discretos e, ainda, com uma combinação destes.
- Não demandam formulações complexas ou reformulações para o problema.
- Não há restrição quanto ao ponto de partida dentro do espaço de busca da solução.
- Possibilita a realização de buscas simultâneas no espaço de possíveis soluções por meio de uma população de indivíduos.

- Otimizam um grande número de variáveis, desde que a avaliação da função objetivo não tenha um custo computacional demasiadamente alto.

A ênfase com relação aos custos computacionais faz-se necessária, pois a maior desvantagem apresentada com relação aos métodos estocásticos diz respeito ao custo computacional que alguns métodos necessitam, pois as técnicas trabalham com um conjunto de pontos onde necessitam de muitas iterações da função objetivo para encontrar o resultado esperado.

Os Métodos Estocásticos que destacamos são:

- Simulated Annealing (SA) (*KIRKPATRICK; GELATT,* 1983).
- Particle Swarm Optimization (PSO) (KENNEDY; EBERHART, 1995).
- Luus-Jaakola (LJ) (LUUS; JAAKOLA, 1973).
- Particle Collision Algorithm (PCA) (SACCO; OLIVEIRA, 2005).
- Alcateia (CORRÊA JR.; CORRÊA, 2016).

Cada um desses métodos será analisado nos tópicos que se seguem, com uma maior ênfase ao método Alcateia, por ser este objeto do presente estudo.

2.1.2.1 Método Simulated Annealing (SA)

O método Simulated Annealing (SA) é também conhecido como recozimento simulado. O método foi proposto por Kirkpatrik e Gellat (apud WEI et al., 2018) e é baseado no processo de recozimento de metais,

Neste método os metais são aquecidos acima de seu ponto de fusão e, lentamente, são resfriados. Desse modo, se reorganizam em estruturas cristalinas de mínima energia. Segundo Viana (2017), o termo recozimento é utilizado para definir "[...] o processo de resfriamento térmico iniciado com a liquidificação de um cristal em alta temperatura e seguido de uma lenta diminuição da temperatura para a sua solidificação".

O objetivo de diminuir, lentamente, a temperatura é conseguir alcançar a solidificação apenas quando o sistema estiver em seu estado mínimo de energia[1]. Este estado não seria alcançado se a diminuição de temperatura ocorresse de maneira brusca (LARA HERRERA, 2007).

[1] Estados mínimos de energia se caracterizam por uma perfeição estrutural do material utilizado no recozimento.

Segundo Eglese (1990, p. 127): “[...] em métodos de busca local clássicos, a partir da solução atual é gerada uma nova solução tentativa e esta substitui a anterior se o valor da função objetivo (energia) associada for menor”.

No caso do uso de técnicas de recozimento simulado, permite-se transições em busca de soluções de maior energia. Mas para tanto é necessário que haja certa probabilidade para evitar que o método fique preso na bacia de atração de um mínimo local. Esta probabilidade diz respeito à temperatura do sistema físico original e, com o passar das iterações, diminui (VIANA, 2017).

2.1.2.2 Método Particle Swarm Optimization (PSO)

O método Particle Swarm Optimization (PSO) é também conhecido como otimização por enxame de partículas, foi proposto por Kennedy e Eberhart (1995).

As técnicas de enxame proposta pelos estudiosos (KENNEDY; EBEHART, 1995) exploram a analogia com o comportamento social de animais, como: enxame de abelhas, cardumes de peixes ou bando de pássaros. Nestes grupos de animais, cada indivíduo do grupo toma decisões próprias, mas, de certa forma, sempre se baseando nas experiencias do líder do grupo.

Matematicamente falando, considera-se cada indivíduo do bando como um ponto do espaço n-dimensional e a velocidade do animal, a direção de busca usada nesta busca candidata a solução, pois “[...] a direção da busca em uma iteração é determinada através da ponderação entre a experiência daquela solução e da melhor solução já encontrada pelo grupo (metaforicamente, a solução líder” (PAPADAKIS; MARKAKI, 2019).

Este método, segundo Papadakis e Markaki (2019), tem base na simulação de um modelo de iteração social simplificado. Neste modelo, cada indivíduo da população tem a capacidade de avaliar tanto a qualidade da sua própria experiência como a qualidade das experiências dos outros membros da população e utilizá-la na resolução de problemas propostos.

Dessa forma, a tomada de decisão de cada indivíduo dependerá tanto do seu desempenho no passado quanto do desempenho no passado dos outros membros da população. Essas experiências correspondem à aprendizagem individual (cognitiva) e à transmissão cultural (social) (SERAPIÃO, 2009).

2.1.2.3 Método Luus-Jaakola (LJ)

O método estocástico Luus-Jaakola (LJ) foi inicialmente apresentado pelos autores Rein Luus e T. H. I. Jaakola (1999 apud AL-MARZOUG; HODGSON, 2006) e é um método que tem como base: buscas aleatórias que utilizam apenas as informações da função objetivo a ser otimizada.

Este é, dentre os demais métodos estocásticos de busca aleatórios, o algoritmo mais popular. Inicialmente ele foi proposto para a resolução de problemas de engenharia química (LOBATO et al., 2007). No entanto, nos dias atuais, o algoritmo tem sido aplicado em diversas áreas, dentre elas: problemas de controle ótimo com arcos singulares (SILVA NETO; SOEIRO, 2012), estimação de propriedades radiativas (KNUPP et al., 2017), resolução de funções matemáticas multimodais (COELHO et al., 2017), dentre outras aplicações.

Este método possui uma excelente eficiência computacional e encontra resultados de qualidade satisfatória. Por esses motivos, “[...] o método LJ vem sendo utilizado com sucesso em problemas de otimização não linear de pequenos e médios portes” (SILVA, 2009).

A ideia básica do método LJ é gerar soluções aleatórias dentro de uma região de busca que é reduzida de tamanho ao longo das iterações. Para isso, o método LJ utiliza os parâmetros de raio de busca, quantidade de laços internos, quantidade de laços externos e coeficiente de contração do raio de busca (VIANA,2017).

Luss e Hennessy (1999) propuseram um procedimento de multi-passos em que a região de busca de cada variável do projeto fosse atualizada de acordo com o valor calculado entre a passagem do laço interno para o externo. No entanto, segundo Jezowsk et al. (2015), mais recentemente, tem sido desenvolvida uma abordagem mais sofisticada que visa reduzir o espaço de busca por meio de atualização de coeficientes de contração em cada looping do algoritmo.

De acordo com Lobato de Steffen (2010), nos últimos anos muitos esforços tem sido promovidos no sentido de aumentar o desempenho do algoritmo Luus-Jaakola. Para tanto, têm sido desenvolvidas metodologias mais eficientes que buscam reduzir o espaço de busca.

2.1.2.4 Método de Particle Collision Algorithm

O Algoritmo de Colisão de Partículas (do inglês Particle Collision Algorithm – PCA) foi desenvolvido no ano de 2005 por Wagner F. Sacco (2005 apud MARTÌNEZ GONZÁLEZ et al., 2014) e tem como base o algoritmo de Simulated Annealing (SA) (KIRKPATRIK et al., 1993), associado a fenômenos físicos de absorção (absorption) e espalhamento (scattering) que ocorrem em reatores nucleares, mais precisamente, no processo de colisão das partículas nucleares, onde a partícula é espalhada por um núcleo-alvo e absorvida pelo núcleo de destino (SACCO; OLIVEIRA, 2005).

Ensinam Kirkpatrik et al. (1993, p. 192) que

> O PCA é inicializado com a seleção (randômica ou não) de um projeto inicial, denominado Old Config, que é modificado estocasticamente pelo operador Perturbação. Este operador permite a obtenção de uma nova solução candidata a ótimo (New Config). Essa nova solução é comparada com a solução atual em termos do valor da função objetivo, que pode ou não ser aceita. Se essa nova solução não é aceita, um esquema do tipo Metropolis é usado com a aplicação do operador Espalhamento. A exploração das vizinhanças da solução candidata é garantida através dos operadores Perturbação e Pequena-Perturbação.

O algoritmo de colisão de partículas possui uma quantidade menor de parâmetros a serem configurados, em comparação a algoritmos clássicos como Algoritmos Genéticos, por exemplo, pois a maioria dos parâmetros são definidos por rotinas aleatórias, como bem colocado por Sacco e Oliveira (2005).

Esta é uma característica que tem tornado o PCA uma abordagem mais atrativa, pois exige a atribuição de um número máximo de gerações. Esta característica tem se mostrado importante, pois este parâmetro influencia, decisivamente, no número de avaliações da função objetivo (LOBATO et al., 2010).

Mais tarde, Luz et al. (2008, p. 77) propuseram “[...] o Multi-Particle Collision Algorithm (MPCA), que consiste na extensão do algoritmo original, concebido para uma única partícula, para M partículas”. Na proposta dos pesquisadores, divide-se para cada partícula o número máximo de gerações.

O que se verificou é que os resultados obtidos se mostraram promissores, tendo em vista que se conseguiu alcançar a mesma quantidade de solução sem que houvesse um aumento no número de parâmetros do algoritmo natural. Estes resultados foram obtidos também por Lobato e Steffen Jr. (2010) ao aplicarem esta abordagem em problemas de controle ótimo com índice flutuante.

Capítulo 3

3 Método de Otimização Estocástico Alcateia

O método Alcateia foi proposto por Corrêa Jr. e Corrêa (2016). O método tem como base o padrão comportamental dos lobos durante o seu processo de caça. As alcateias possuem uma estrutura bem definida, existindo um casal de lobos líderes denominados de lobos alfas.

Durante a caça o lobo alfa escolhe a presa a ser atacada. Para essa escolha ele visa ou o animal mais velho ou o mais novo ou aquele que está adoentado, ou seja, a presa ótima. Além disso, os lobos alfas são os primeiros a se alimentarem e os únicos que podem se reproduzir. Quando outros lobos pretendem tomar a posição do lobo alfa na alcateia, há uma batalha entre eles, sendo considerado o novo lobo alfa aquele que vence essa batalha (CORRÊA JR.; CORRÊA, 2016).

Essas caraterísticas comportamentais são à base do algoritmo Alcateia. A cada iteração do algoritmo, o lobo alfa será aquele que vai encontrar a presa mais acessível, ou seja, aquele que possuir a melhor solução daquela iteração. Além disso, todos os outros lobos tendem a atacar a presa escolhida pelo lobo alfa, isso é caracterizado no algoritmo através do coeficiente de independência, que faz com que os outros lobos convirjam para a solução do problema, refinando a solução encontrada a cada iteração (CORRÊA JR.; CORRÊA, 2016).

A ideia básica do algoritmo Alcateia é que iniciando com uma ampla região de busca do domínio da função objetivo, várias possíveis soluções (as presas), sejam selecionadas e que depois essa região de busca seja reduzida a cada iteração fazendo que essas possíveis soluções convirjam para a solução ótima do problema.

Essa redução da região de busca é representada no algoritmo pela diminuição do número de laços internos conforme são executados os laços externos. No padrão comportamental dos lobos, os laços externos representam o posicionamento dos lobos no cerco da presa enquanto os laços internos representam os lobos indo em direção à presa (CORRÊA JR.; CORRÊA, 2016)

Uma das vantagens do algoritmo Alcateia é sua capacidade de resolução de problemas com inúmeros mínimos locais, pois muitos problemas de modelagem possuem, como uma das suas dificuldades, a existência de muitos mínimos locais. Esse fato faz com que grande parte dos algoritmos de otimização seja excessivamente dependente das condições iniciais para a garantia da sua convergência para o ótimo global (CORRÊA; CORRÊA JR., 2017).

Outras vantagens que podem ser citadas são as seguintes:

- A não dependência do chute inicial para que se encontre o ótimo global da função objetivo.
- A menor frequência em que ótimos locais são apontados como ótimos globais, diferente dos métodos determinísticos, muito dependentes do gradiente.
- A possibilidade de trabalhar com funções não-diferenciais, descontínuas, não-lineares ou multimodais.

No entanto, o uso deste método traz também desvantagem, que é o alto custo computacional (uma das desvantagens mais relatadas na literatura), em se comparado aos dos métodos determinísticos.

Dentre os métodos heurísticos (estocásticos) existe uma classe de heurísticas mais generalizadas, e a estas heurísticas dão-se o nome de meta-heurísticas. Segundo André R. Gonçalves são “[...] abordagens que buscam tratar de diversos tipos de problemas sem a necessidade de realizar grandes alterações na estrutura do algoritmo. Esta metodologia inclui a manipulação de uma ou várias outras heurísticas”.

Gonçalves (2011, p. 57-58) ainda diz que:

> [...] uma meta-heurística pode manipular uma única candidata a solução, parcial ou completa, ou ainda uma coleção de candidatas a soluções a cada iteração. Alguns dos algoritmos que se encaixam na denominação de meta-heurística são, Busca Tabu, Algoritmos de Colônia de Formigas, Enxame de Partículas (Particle Swarm Optimization), Algoritmos Evolutivos, Recozimento Simulado (Simulated Annealing), e suas hibridizações.

Há que se ressaltar que a criação do algoritmo Alcateia teve como intuito criar um método de Otimização Global Estocástico. Ele se mostra, como já visto, muito mais vantajoso do que os métodos determinísticos, no que compete, à resolução de

funções mais complexas. Afinal, uma de suas vantagens é a meta-heurística que os algoritmos populacionais proporcionam.

A característica comportamental do lobo alfa em se basear em escolhas plausíveis para a captura da presa com o enfoque no sucesso da caça é à base do algoritmo. Durante a execução do algoritmo Alcateia, os lobos a cada iteração estão suscetíveis a evoluções hierárquicas, ou seja, a cada iteração será nomeado como Lobo Alfa aquele que houver encontrado a melhor solução durante aquela iteração. Além disso, há o coeficiente de independência que faz com que os outros lobos tendam ou convirjam para o ponto no qual o lobo alfa está presente, refinando assim a cada iteração a solução encontrada até chegar à solução ótima global do problema proposto.

O número de laços internos é diminuído conforme são executados os laços externos (determinação da região de busca).

3.1 Algoritmo do método Alcateia

A seguir, no algoritmo 1 é apresentado o pseudocódigo para o Alcateia utilizado nesse trabalho.

Algoritmo 1 – Alcateia

```
1  Escolha o número de iterações externas Le
2  Escolha o número de iterações internas Li
3  Escolha o número de lobos NLobos
4  Escolha o coeficiente de contração c
5  Escolha a constante de independência Id
6  Gerar aleatoriamente a solução x0
7  Gerar aleatoriamente os passos delta
8  Para i de 1 até Le faça
9      aux ← x0          // variável auxiliar
10     Para j de 1 até Li faça
11         Para k de 1 até NLobos faça
12             lambda ← (-1+2*rand())
13             x[:,k]  ←  x0[:,k] + lambda *.delta[:,k]
14             g[k]  ← f(x0[:,k])
15             Se f(x[:,k]) < g[k] ) então
16                 x0[:,k]  ←   x[:,k]
```

```
17                      g[k]  ← f(x[:,k])
18                  Fim se
19              Fim para
20          Fim para
21          [fxalfa  p] ← min(g)  //retorna o menor valor da função g e sua posição.
22          Xalfa  ←  x0[:,p]
23          Para t de 1 até NLobos faça
24              x0[:,t]  ← (Id * x0[:,t] + (1-d)* Xalfa)
25              delta[:,t] ← abs(x0[:,t] – aux[:,t])
26          Fim para
27          Li ←  Li *(1-c)
28  Fim para
```

3.1.1 Etapas do Algoritmo Alcateia

O algoritmo Alcateia contém cinco etapas:

- posicionamento de cada lobo em seu espaço de busca;
- escolha do lobo alfa;
- convergência dos lobos para a região do lobo alfa;
- atualização do raio de busca e
- contração do laço interno.

A função Ackley é utilizada, neste trabalho, para exemplificar estas cinco etapas do algoritmo. Esta função é amplamente usada para testar algoritmos de otimização. Seu formato, como mostrado na Figura 3.1 é caracterizado por uma superfície exterior quase plana, formadas por pequenas concavidades e um grande buraco no centro. Essa topologia representa uma dificuldade aos métodos de otimização, por ficarem presos nos vários mínimos locais (SURJANOVIC; BINGHAM, 2013). A equação da função Ackley dada por Surjanovic e Bingham (2013) é a seguinte:

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{3}\sum_{i=1}^{3} x_i^2}\right) - \exp\left(\sqrt{\frac{1}{3}\sum_{i=1}^{3}\cos(2\pi x_i)}\right) + 20 + e^1 \qquad (3.1)$$

Com o raio de busca -32,768 ≤ x_1 ≤ 32,768 e -32,768 ≤ x_2 ≤ 32,768.

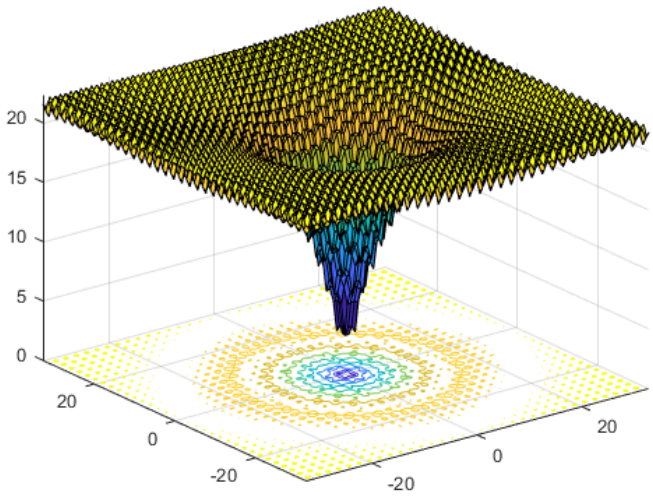

Figura 3.1: Resultado de simulação da função Ackley.

3.1.1.1 Posicionamento de cada lobo no seu espaço de busca

Para iniciar o método, escolhe-se o espaço de busca, delta, o número de laços externos, Le, o número de laços internos, Li, o número de lobos, Nlobos, o coeficiente de contração, c, e o coeficiente de independência, Id, e a geração da população inicial (lobos), x0.

Durante cada iteração do laço interno, processo de caça, cada lobo é atualizado conforme indicado abaixo:

$$\text{lambda} \leftarrow (-1+2*\text{rand}()); \tag{3.2}$$

$$x[:,k] \leftarrow x0[:,k] + \text{lambda} *.\text{delta}[:,k], \tag{3.3}$$

onde lambda é um vetor de números aleatórios entre -1 e 1. E o produto termo-a-termo entre os vetores lambda e delta, resulta no passo do lobo, que somado com a posição de referência, x0, tem-se uma candidata a uma nova posição. Para cada candidata a uma nova posição, o valor do seu funcional é avaliado e, caso um melhor valor seja encontrado, esta se torna a nova posição do lobo.

Para detalhar melhor a etapa, foram usados os valores dos parâmetros da Tabela 3.1. Com o número de laços externo igual a 1 e o número de laço interno igual a 20, o total de iterações é igual a 20.

A Figura 3.2 ilustra o processo de atualização da posição de um lobo. Partindo do ponto circulado em vermelho, nas 11 primeiras iterações, serão testadas 11 diferentes posições. Estas, testadas na função objetivo, apenas apresentaram valor melhor que a posição de partida na última iteração, marcando a atualização da posição do lobo para o ponto circulado em azul.

Tabela 3.1: Valores dos parâmetros da função Ackley.

Parâmetro	**Valor**
Número de variáveis	2
Número de Lobos	1
Número de Laços Externos	1
Número de Laços Internos	20
Coeficiente de Independência	0,8
Coeficiente de Contração	0

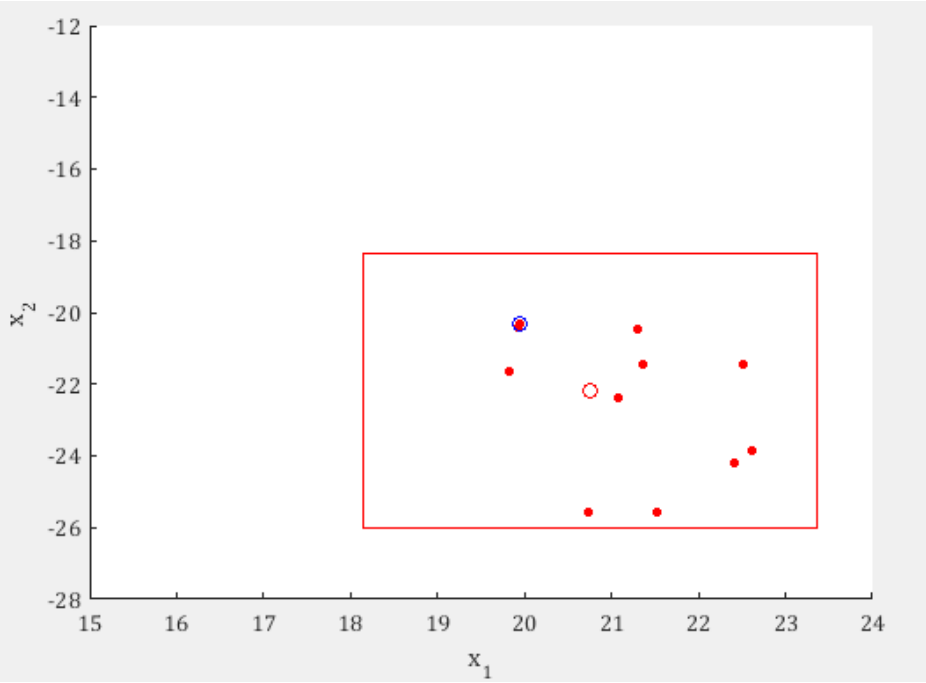

Figura 3.2: Primeira atualização da posição do lobo.

A Figura 3.3, em continuação, são testadas mais 5 posições antes da segunda atualização de posição.

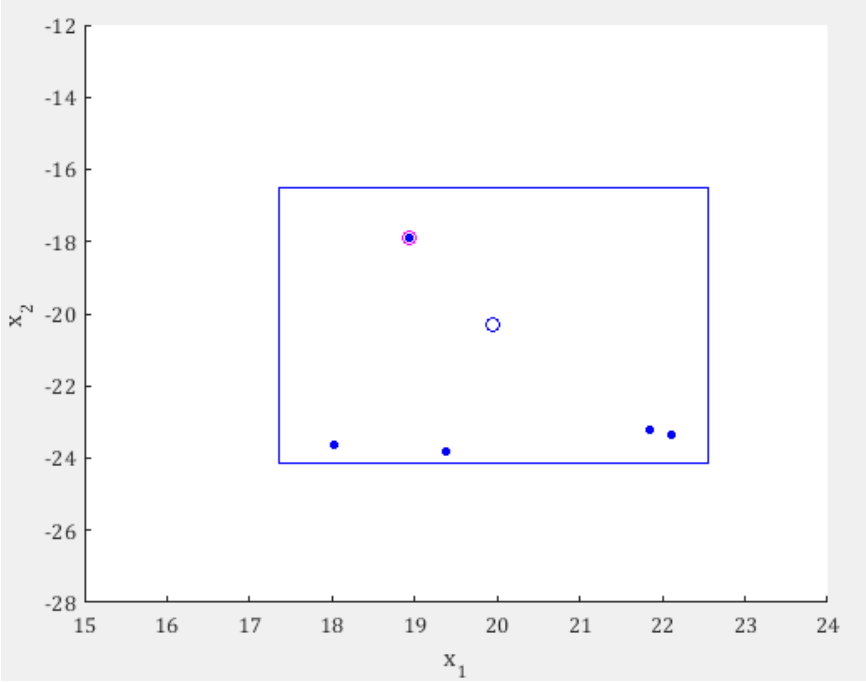

Figura 3.3: Segunda atualização da posição do lobo.

Na Figura 3.4 são testadas mais 3 posições para a terceira atualização de posição. E na Figura 3.5 não há atualização da posição do lobo, sendo o ponto encontrado na Figura anterior (Figura 3.4) o melhor valor para a função objetivo em 20 iterações encontrado pelo método.

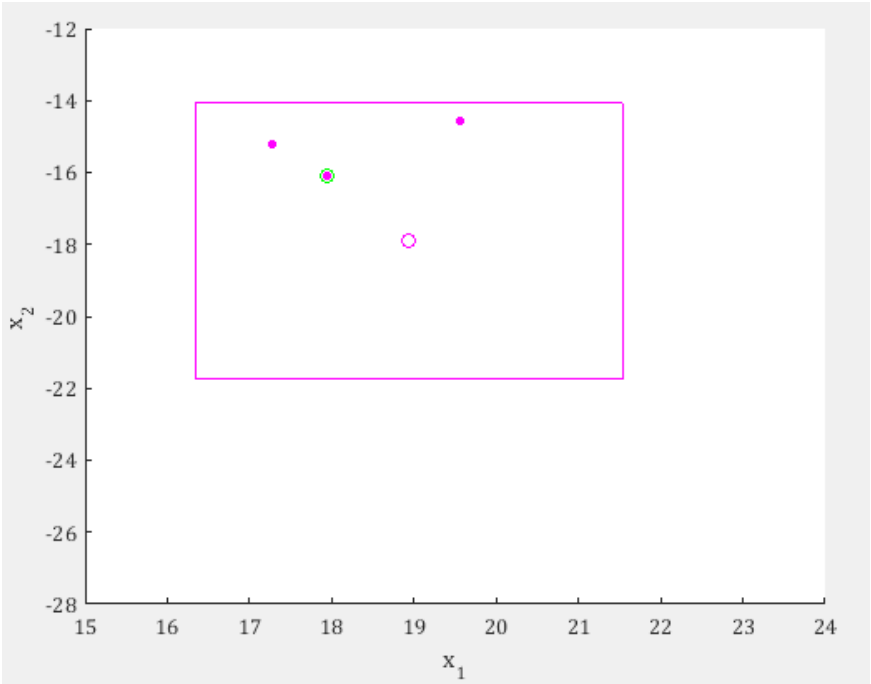

Figura 3.4: Terceira atualização da posição do lobo.

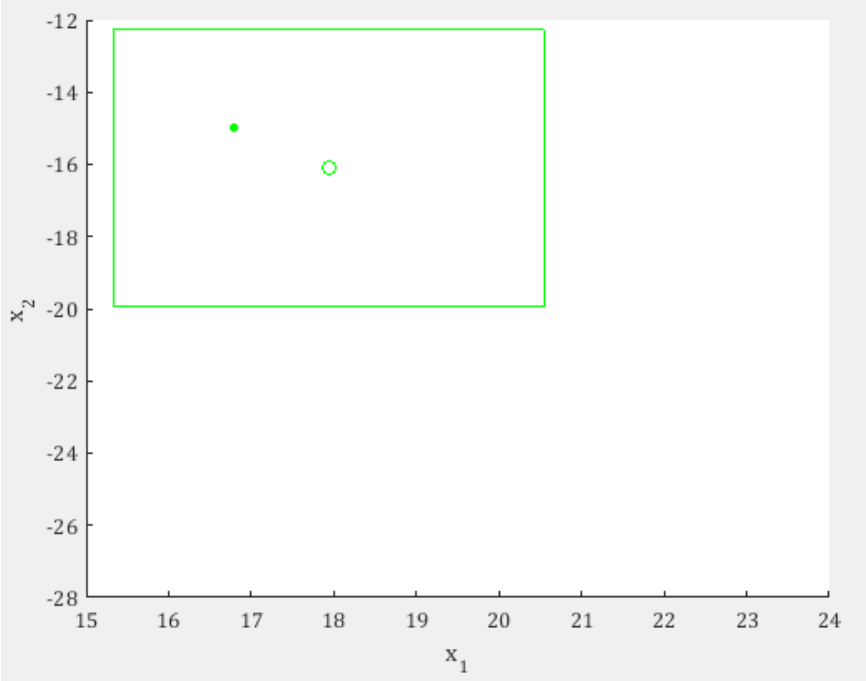

Figura 3.5: Posição final do lobo.

Na figura 3.6 tem todas as posições testadas. As atualizações de posições do lobo estão circuladas.

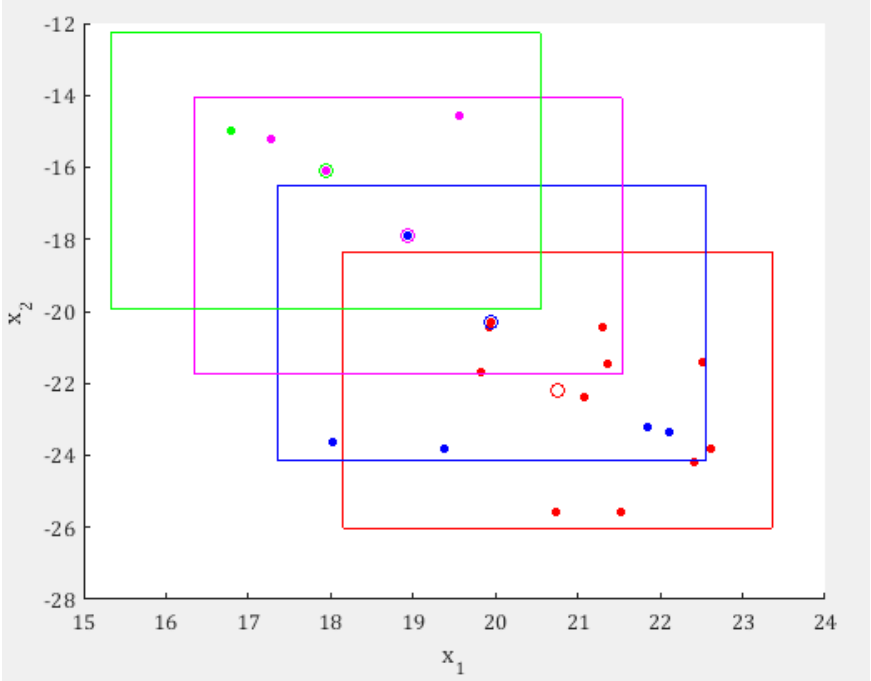

Figura 3.6: Todas as atualizações de posições do lobo nas 20 iterações.

Este é o processo de atualização da posição de apenas um lobo. Caso na Tabela 3.1 tivesse mais lobos, este processo repetiria para cada lobo.

3.1.1.2 Escolha do lobo alfa

Depois que todos os laços internos do primeiro laço externo forem executados, tendo como valores de parâmetros a Tabela 3.2, seleciona-se o ponto que representa o melhor candidato a solução ótima da função objetivo. A Figura 3.7 ilustra 15 lobos disperso no espaço de busca da Alcateia, representados por quadrados de cores diferentes. O lobo alfa, localização do melhor candidato a solução ótima, foi marcado com uma cruz (+).

Tabela 3.2: Valores de parâmetros para escolha do lobo.

Parâmetro	**Valor**
Número de variáveis	2
Número de Lobos	15
Número de Laços Externos	2
Número de Laços Internos	10
Coeficiente de Independência	0,8
Coeficiente de Contração	0

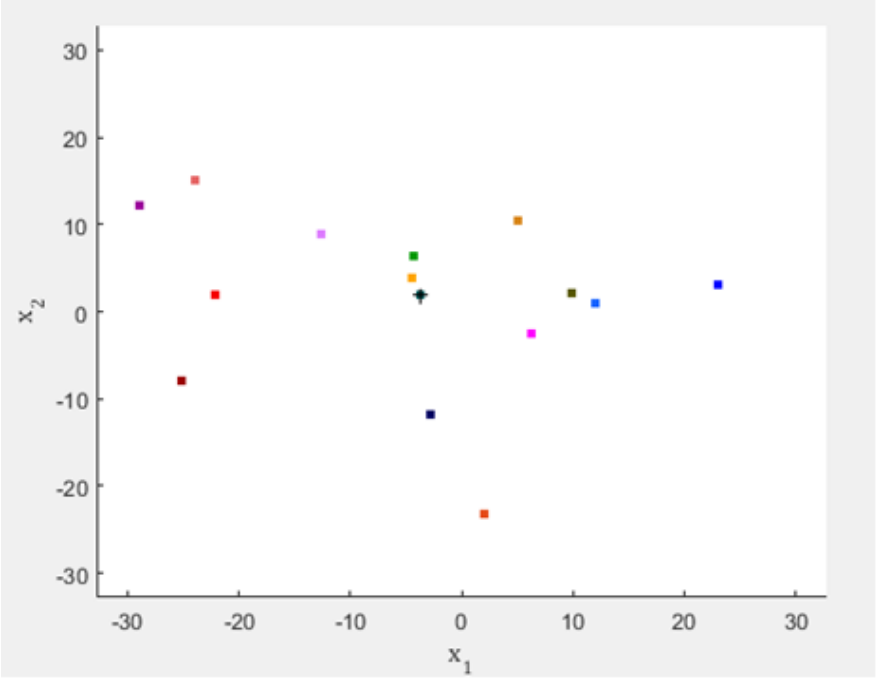

Figura 3.7: Lobo alfa na primeira execução do laço externo.

A Figura 3.8 mostra a escolha de um novo lobo alfa, depois que todos os laços internos do segundo laço externo foram executados.

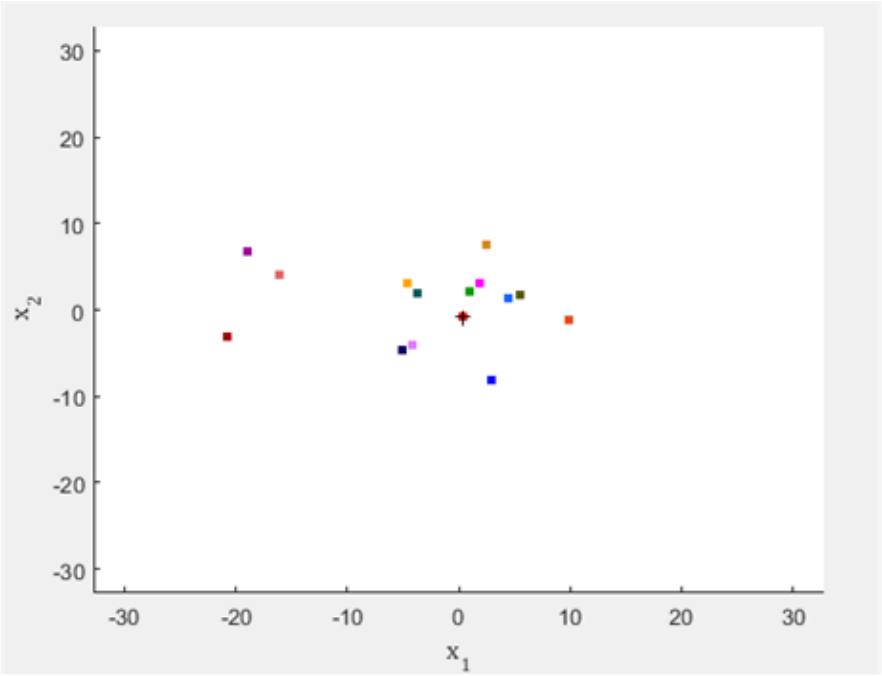

Figura 3.8: Lobo alfa na segunda execução do laço externo.

3.1.1.3 Convergência dos lobos para a região do lobo alfa

Com a escolha do lobo alfa, a execução do algoritmo passa para uma nova etapa que é a convergência dos lobos para a região do lobo alfa. Esta alteração da posição de cada lobo vai depender da posição do lobo alfa, *Xalfa*, e o coeficiente de independência, *Id*, segundo a Equação (3.4).

$$x[:,t] \leftarrow (Id * x0[:,t] + (1-Id)* Xalfa) \tag{3.4}$$

A Figura 3.9 ilustra 14 lobos e o lobo alfa antes do processo de convergência regido pela Equação (3.4). As Figuras 3.10, 3.11 e 3.12 retratam o momento seguinte, após o processo de convergência, com coeficiente de independência 1, 0,6 e 0,3, respectivamente. A observância das figuras permite notar a influência do coeficiente de independência no processo de convergência. Sendo esta, inversamente proporcional ao Id.

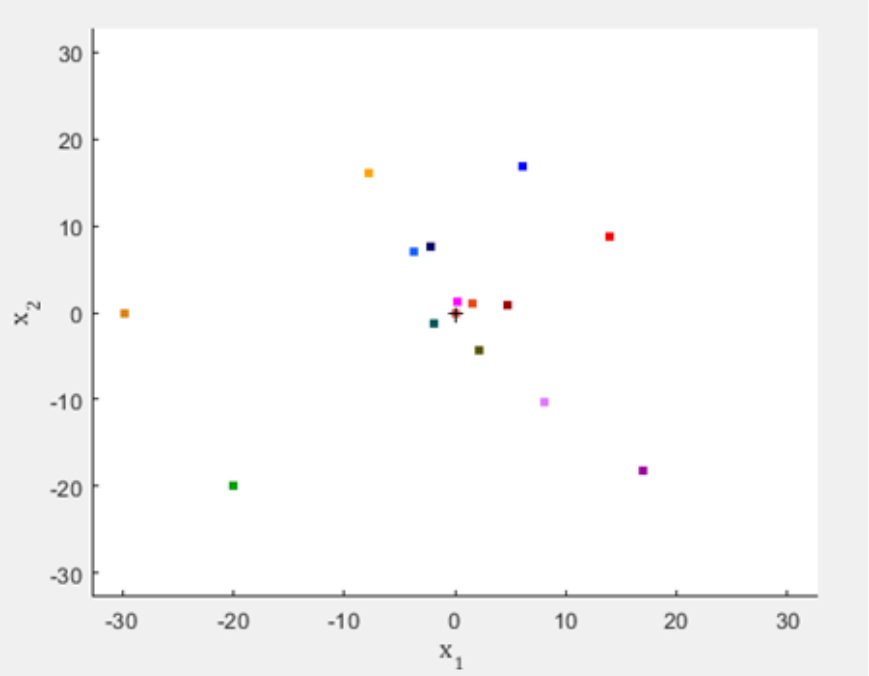

Figura 3.9: Antes da convergência.

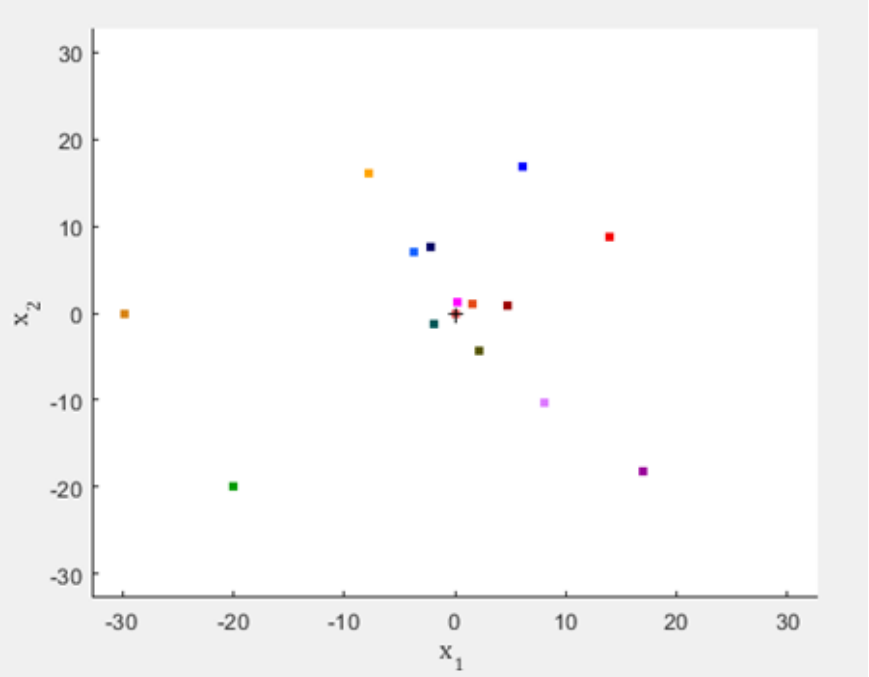

Figura 3.10: Convergência com coeficiente de independência igual a 1.

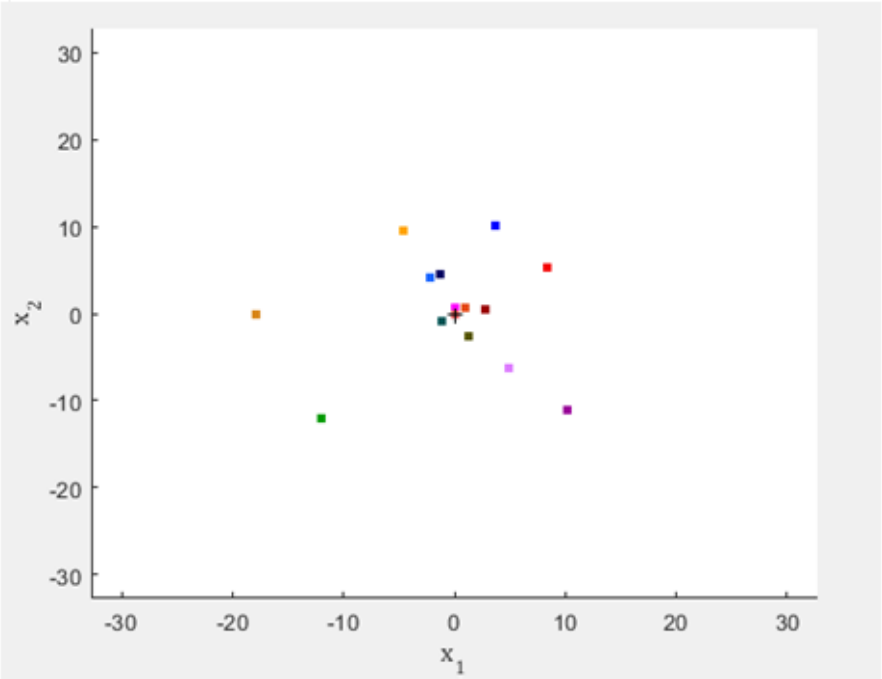

Figura 3.11: Convergência com coeficiente de independência igual a 0,6.

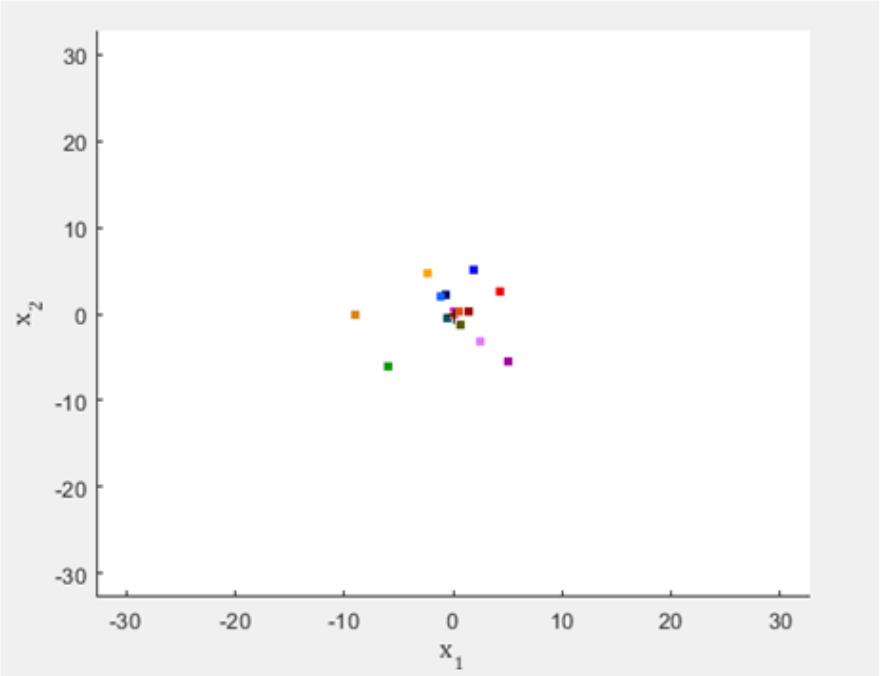

Figura 3.12: Convergência com coeficiente de independência igual a 0,3.

Quanto menor coeficiente de independência, maior é a convergência. Com isto tem-se uma redução na área de busca da alcateia.

3.1.1.4 Atualização do raio de busca do lobo

Depois da convergência, a execução do algoritmo passa para a etapa que é a atualização do raio de busca, *delta*, de cada lobo. O valor do delta é o deslocamento do lobo, isto é, a diferença da posição atual menos a posição anterior (Equação (3.5)).

$$delta[:,t] \leftarrow abs(x0[:,t] - aux[:,t]) \tag{3.5}$$

Na Figura 3.13 tem-se que posição inicial do lobo com marcação azul (1,2) e a posição final como marcador vermelho (2,4). Então o lobo deslocou de 1 unidade no eixo horizontal, de 1 para 2, e no eixo vertical deslocou 2 unidades, de 2 para 4. Logo tem-se um novo delta para este lobo que é igual a (1,2), valor absoluto da diferença da posição atual menos a posição anterior (Equação (3.5)).

A candidata a nova posição do lobo é a posição do lobo mais lambda, valor de 1 à -1 (Equação (3.6)), multiplicada pelo delta (Equação (3.7)). Com isto tem-se uma nova área de busca do lobo (retângulo verde).

$$lambda \leftarrow (-1+2*rand()) \tag{3.6}$$

$$x[:,k] \leftarrow x0[:,k] + lambda *.delta(:,k) \tag{3.7}$$

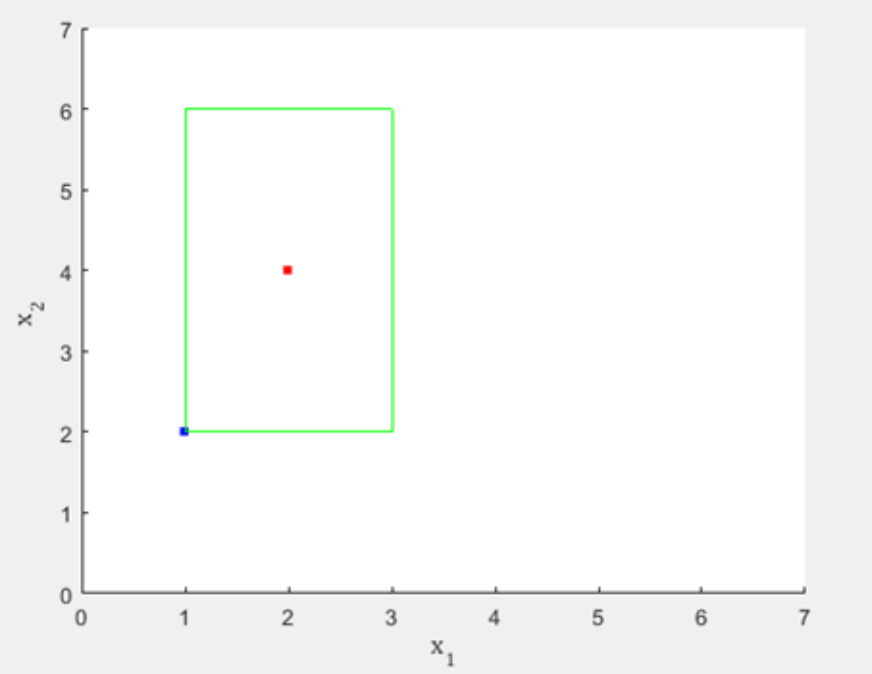

Figura 3.13: Nova posição do lobo.

3.1.1.5 Contração do laço interno

Como a última etapa do algoritmo Alcateia, tem-se a contração do laço interno. Na medida em que que os lobos se convergem para o lobo alfa, diminui o espaço de busca da alcateia. Com isto não tem necessidade de gerar muitas posições candidatas de cada lobo. Logo o número de laços internos, Li, tem uma diminuição percentual de acordo com o coeficiente de contração, *c,* fazendo que a execução do algoritmo gaste menos tempo computacional.

$$Li \leftarrow Li *(1-c) \tag{3.8}$$

De acordo com a Equação (3.8), se o número de laço interno é inicializado com 100, com coeficiente de contração igual a 0,1, na primeira execução do laço externo

serão executadas 100 iterações, na segunda execução do laço externo serão executadas 90 iterações, na terceira 81 iterações e assim sucessivamente.

3.1.2 Tempo de execução

Os hardwares e softwares utilizados para fazerem os testes são os seguintes:

- Processador: intel core i5-4460 CPU @ 3.20GHZ;
- Memória instalada: 8 GB;
- Tipo de sistema: Sistema Operacional de 64 bits;
- Sistema Operacional: Windows 10 Pro;
- Linguagem de Programação: MATLAB R2018A

Para resolver a função Ackley foram usados os seguintes valores paras os parâmetros:

Tabela 3.3: Valores dos parâmetros da Função Ackley.

Parâmetro	**Valor**
Número de variáveis	2
Número de Lobos	15
Número de Laços Externos	25
Número de Laços Internos	100
Coeficiente de Contração	0,1
Coeficiente de Independência	0,3

O método Alcateia encontrou o valor para a função objetivo igual a 8,8818e-16 em 50 vezes em que foi executado. Valor muito bom em comparação ao valor analítico que é igual a 0 (Figura 3.14) em um tempo médio de 0,04369s.

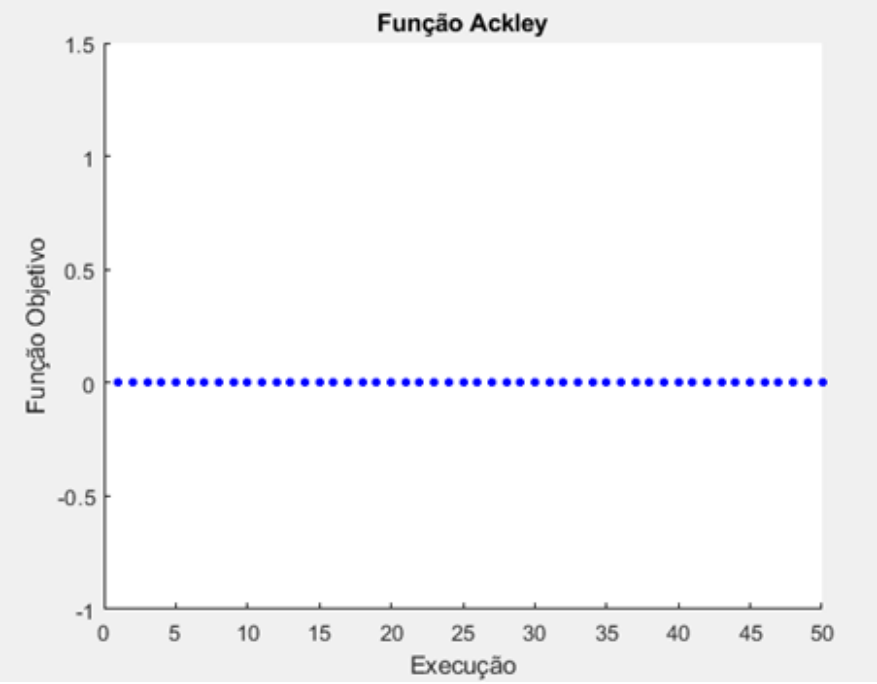

Figura 3.14: Resultados da função Ackley.

Tendo estes valores de parâmetros como ideal para executar a função Ackley, estes deverão ser utilizados como base para análise dos parâmetros do método Alcateia.

3.1.2.1 Tempo de execução em relação a quantidade de lobos

Alterando o valor da quantidade de lobos de 15 para 20 tem-se os mesmos resultados encontrados, função objetivo igual a 8,8818e-16 em 50 vezes em que foi executado, porém o tempo médio de execução foi de 0,04369s para 0,05515s.

Alterando o valor da quantidade de lobos de 15 para 10 não foi encontrado um ótimo resultado para todas as execuções como é mostrado na Figura 3.15. Foram encontrados 47 vezes a valor da função objetivo igual a 8,8818e-16 e 3 vezes o valor igual a 4,4409e-15, porém o tempo médio teve uma redução de 0,04369s para 0,02979s.

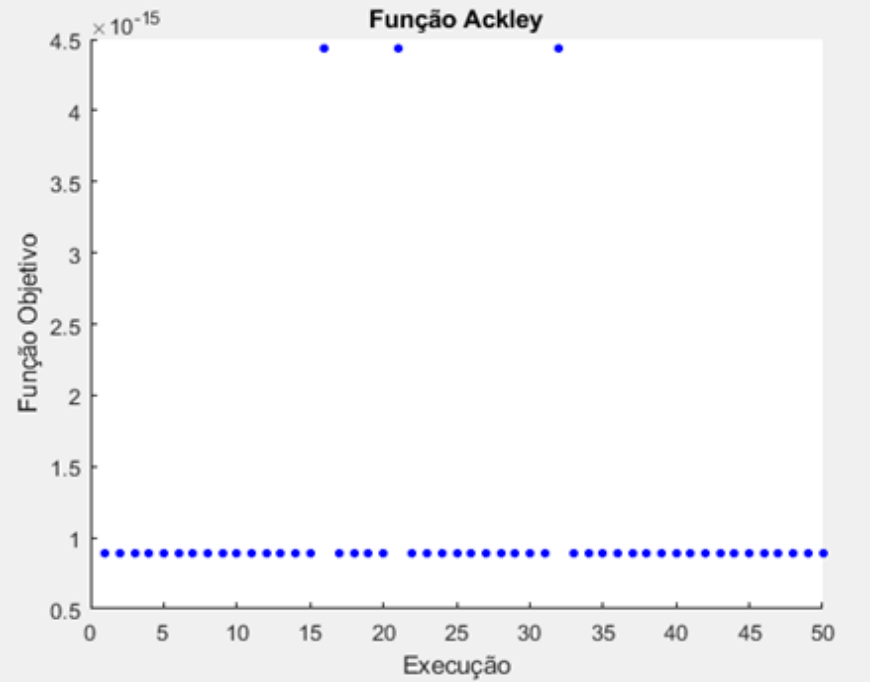

Figura 3.15: Resultado da função Ackley com 10 lobos.

Alterando o valor da quantidade de lobos de 15 para 5, foram encontrados 28 vezes a valor da função objetivo igual a 8,8818e-16(Figura 3.16), com o tempo médio de execução de 0,04369s para 0,01541s.

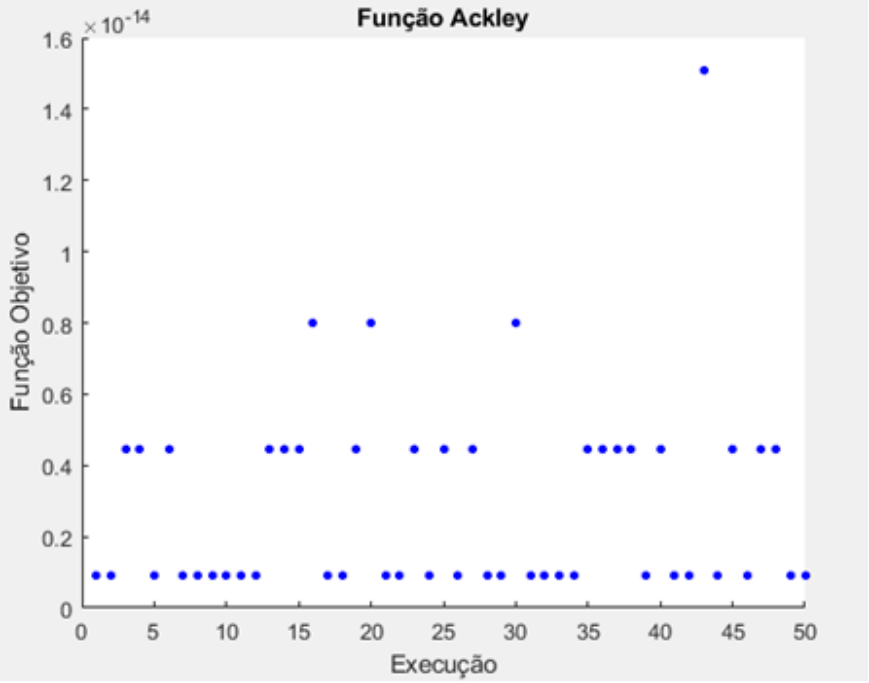

Figura 3.16: Resultado da função Ackley com 5 lobos.

Logo diminuindo a população do método de uma configuração ideal, perde a eficiência e reduz o custo computacional. Analisando o gráfico da Figura 3.17 de uma configuração de 2 até 50 lobos variando de 4 em 4 verifica-se que aumenta o tempo de execução quando é aumentado o número de lobos.

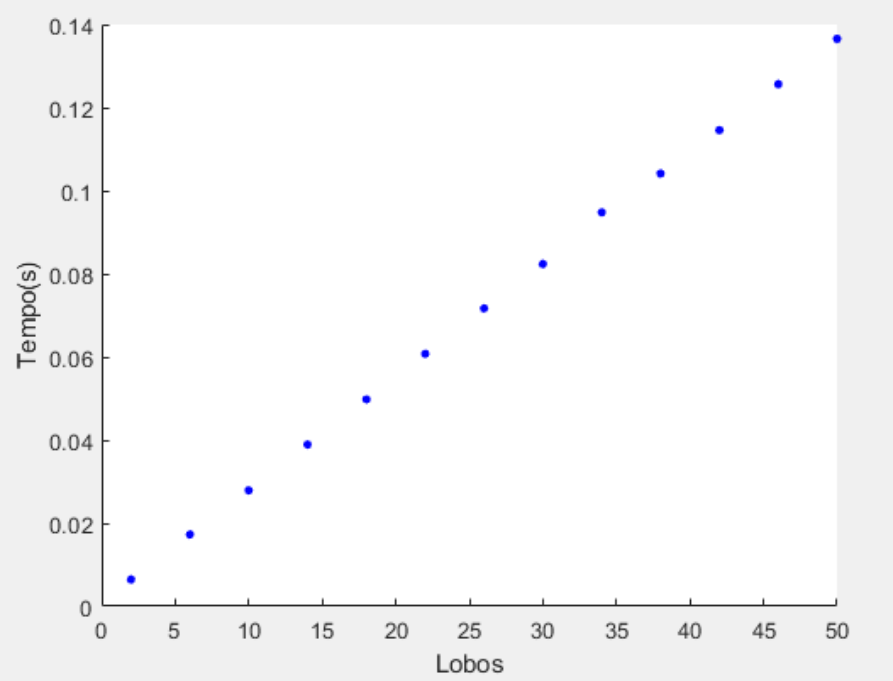

Figura 3.17: Tempo X Quantidade de lobos.

3.1.2.2 Tempo de execução em relação a quantidade de iterações

Tendo como base os valores dos parâmetros da Tabela 3.2 e alterando o valor do laço externo de 25 para 30, tem-se os mesmos resultados para o valor da função objetivo, 8,8818e-16 em 50 vezes em que foi executado, com tempo de execução de 0,04369s para 0,04474s.

Reduzindo o valor do laço externo de 25 para 22, foram encontrados ótimos resultados para todas as execuções como é mostrado na Figura 3.18. Foram encontrados 47 vezes a valor da função objetivo igual a 8,8818e-16 e 3 vezes o valor igual a 4,4409e-15, com o tempo médio de 0,04124s:

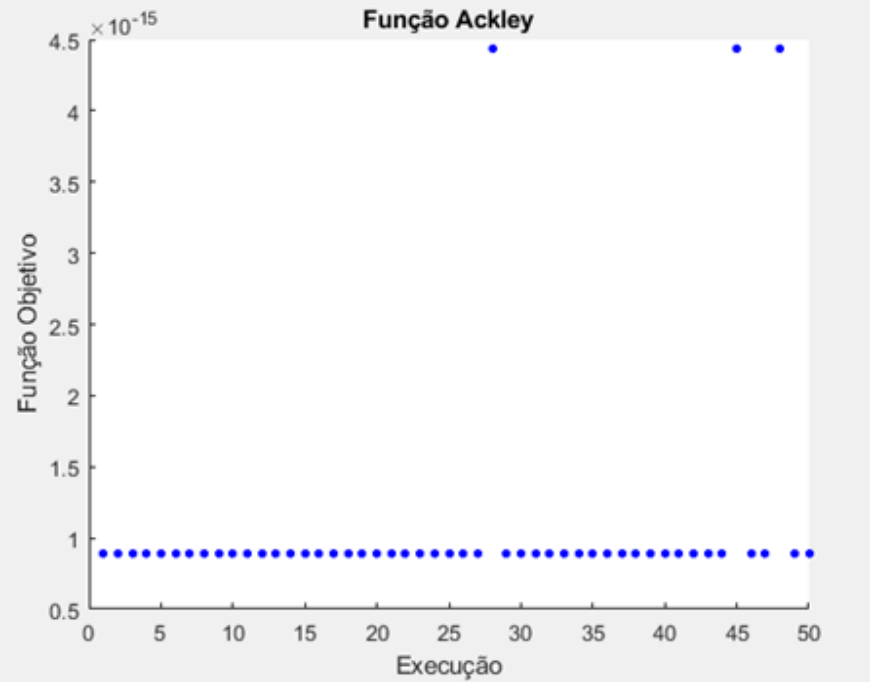

Figura 3.18: Resultado da função Ackley com laço externo igual a 22.

E alterando de 25 para 20 o valor do laço externo, foram encontrados 5 vezes a valor da função objetivo igual a 8,8818e-16 com o tempo médio de 0,04071s em 50 execuções (Figura 3.19).

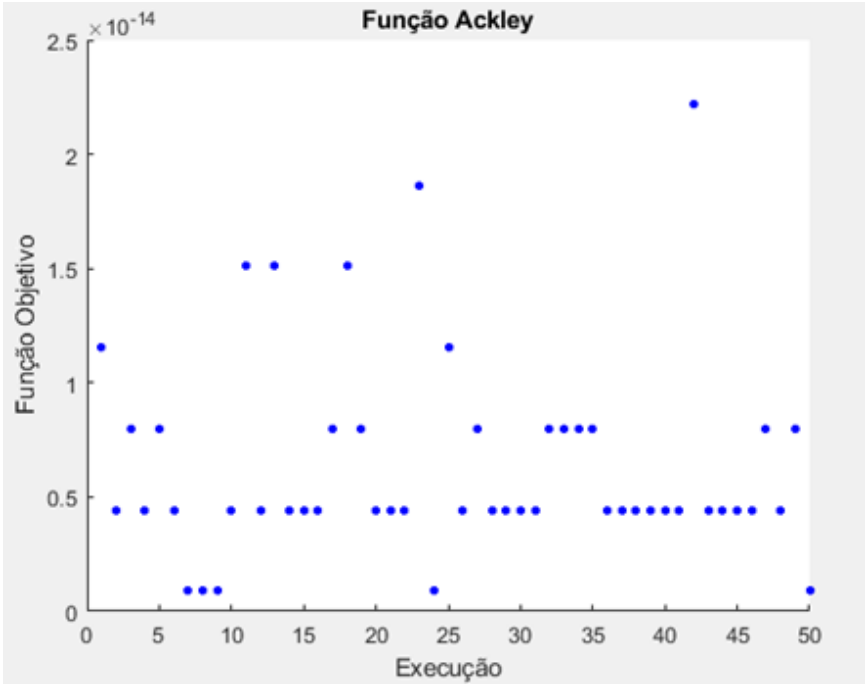

Figura 3.19: Resultado da função Ackley com laço externo igual a 20.

O valor do laço externo influencia muito na precisão do resultado. Porém, quanto maior o valor do laço externo maior é o custo computacional. No gráfico da Figura 3.20, tem-se o valor do laço externo inicializando com 10, variando de 5 em 5, com coeficiente de contração igual a zero, o aumento em todos os tempos de execução é evidenciado.

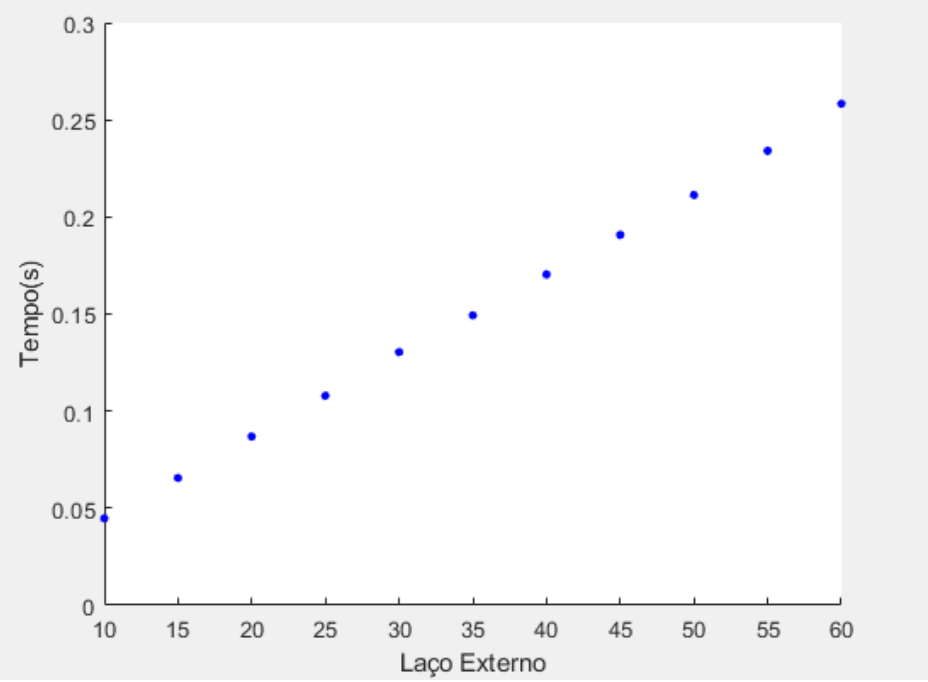

Figura 3.20: Tempo X Laço externo sem contração.

Mantendo a configuração, mas alterando o coeficiente de contração de 0% para 10%, tem-se uma redução do custo computacional para cada valor de laço externo, como pode ser visto comparando o gráfico da Figura 3.20 com o gráfico da Figura 3.21.

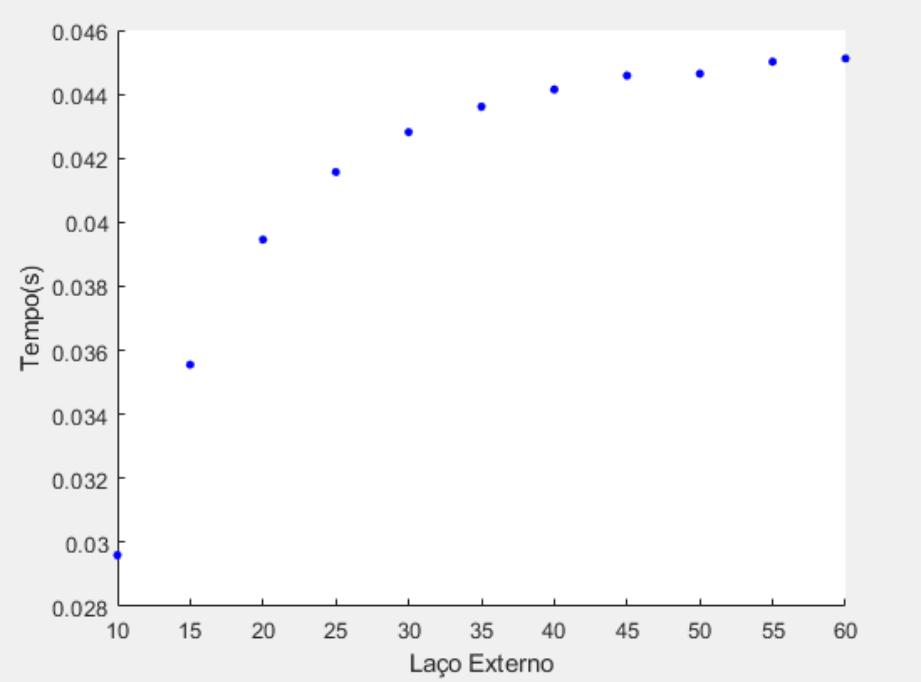

Figura 3.21: Tempo X Laço externo com contração.

Capítulo 4

4 Resultados e discussão

A seguir serão apresentados os resultados do método de otimização Alcateia. Com o propósito de avaliar a eficiência e a robustez, ou seja, a capacidade de localizar ótimos globais com baixo custo computacional e em funções variadas, o método foi aplicado em funções multimodais com e sem restrições, projetos de engenharia e em identificação de danos em uma viga.

4.1 Funções Multimodais irrestritas

As funções multimodais representam o cenário adequado para aplicação do algoritmo Alcateia, devido a sua capacidade de resolver problemas com inúmeros mínimos e máximos locais.

As funções estudadas nesta pesquisa são: Eggholder apresentada por Jamil, M., & Yang, X. S. (2013) e funções Drop-Wave e Griewank apresentadas por Surjanovic e Bingham (2013). Problemas já tratados pela literatura em diferentes trabalhos e que, nesse trabalho, tem seus resultados comparados aos do Alcateia.

4.1.1 Função Eggholder

A função Eggholder, por possuir vários mínimos locais, apresenta elevada dificuldade de otimização. Apresenta um formato de caixa de ovos (Figura 4.1), o que lhe confere o nome.

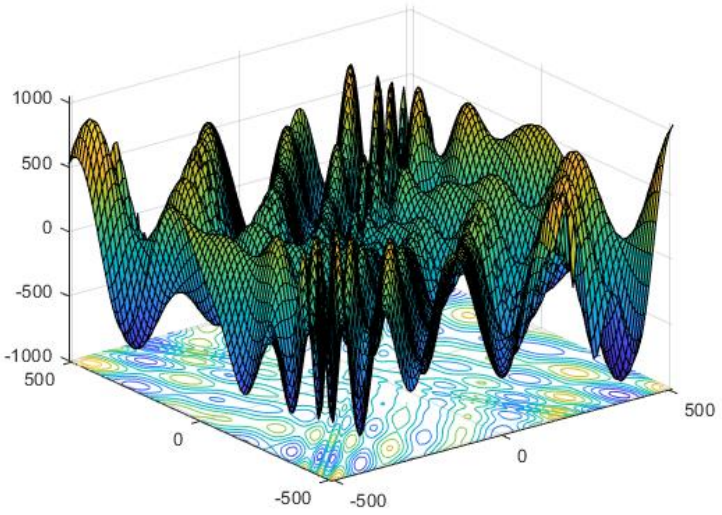

Figura 4.1: Resultado de simulação da função Eggholder.

Sua equação é apresentada por Jamil, M., & Yang, X. S. (2013):

$$f(x) = -(x_2 + 47)\text{sen}\left(\sqrt{\left|x_2 + \frac{x_1}{2} + 47\right|}\right) - x_1\text{sen}\left(\sqrt{|x_1 - (x_2 + 47)|}\right) \qquad (4.1)$$

E tem como o raio de busca -512 ≤ x_1 ≤ 512 e -512 ≤ x_2 ≤ 512.

Para obter um resultado mais consistente foram usados os valores dos parâmetros da Tabela 4.1. Nas 50 execuções o valor da função objetivo foi igual a -959.6407 (Figura 4.2), sendo x_1 igual a 512 e x_2 igual a 404,2319 em um tempo médio de execução de 0,2102s. O valor da função objetivo foi o mesmo encontrado pelo valor analítico. No valor de x_1 o método Alcateia encontrou o valor de 512, ou seja, o mesmo que o valor analítico. Já o valor de x_2 teve uma pequena diferença, ou seja, o método Alcateia encontrou valor igual a 404,2318 enquanto o valor analítico foi igual a 404,2319.

Tabela 4.1: Valores dos parâmetros da função Eggholder.

Parâmetro	**Valor**
Número de variáveis	2
Número de Lobos	15
Número de Laços Externos	32
Número de Laços Internos	1000
Coeficiente de Independência	0,3
Coeficiente de Contração	0,2

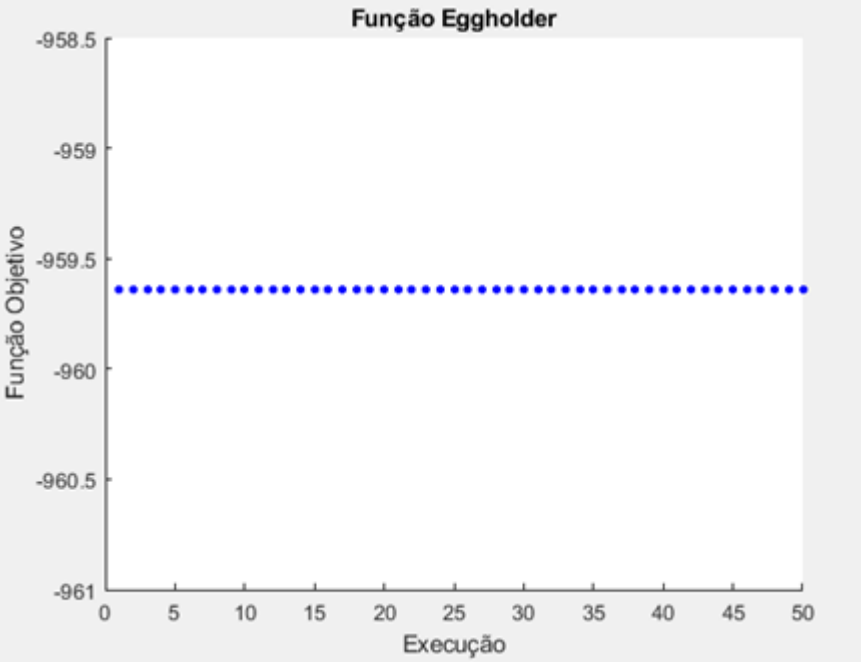

Figura 4.2: Resultados da função Eggholder.

Com intuito de reduzir o tempo de execução foi reduzido o número de lobos de 15 para 8. Com esta alteração o tempo médio de 0,2102s passou para 0,1070s, reduzindo, em mais da metade, o tempo de execução com apenas seis valores diferentes do valor analítico (Figura 4.3).

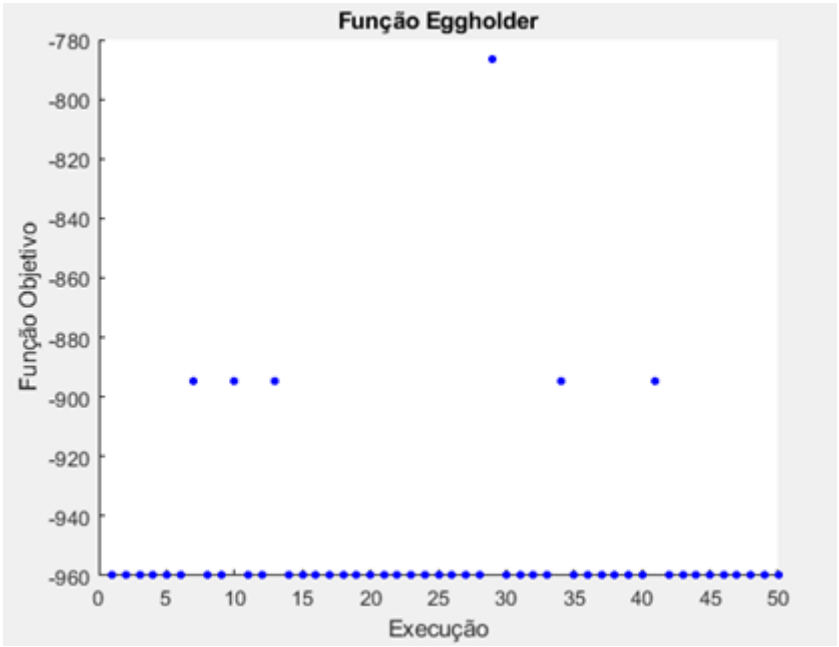

Figura 4.3: Resultados da função Eggholder com redução de lobos.

4.1.2 Função Griewank

A função Griewank possui inúmeros mínimos locais uniformemente distribuídos. Sua forma é mostrada na Figura 4.4. Na Figura 4.5 foi dado um zoom para mostrar a sua característica multimodal.

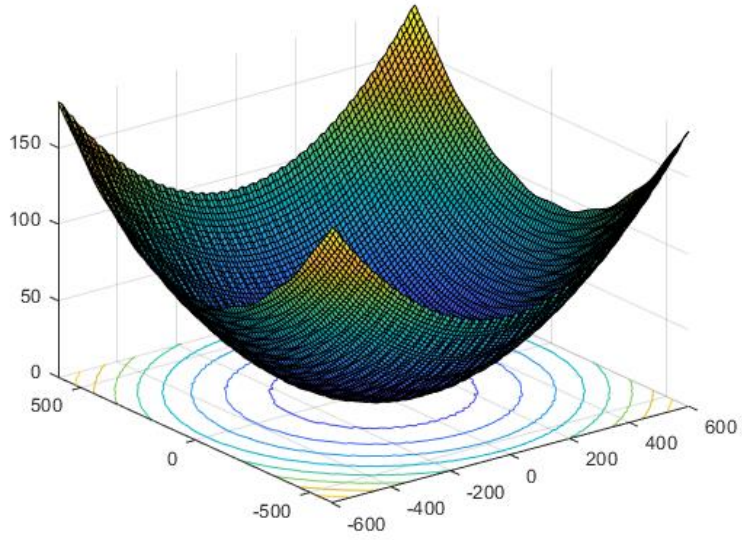

Figura 4.4: Resultado de simulação da função Griewank.

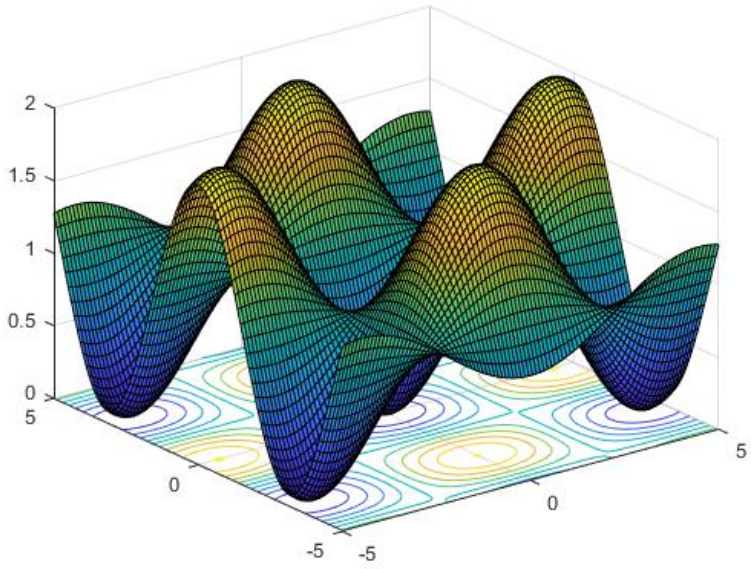

Figura 4.5: Resultado de simulação da função Griewank (centralizada).

A sua equação apresentada por Surjanovic & Bingham (2013):

$$f(x) = \sum_{i=1}^{3} \frac{x_i^2}{4000} - \prod_{i=1}^{3} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad (4.2)$$

Seu raio de busca é de $-600 \leq x_1 \leq 600$ e $-600 \leq x_2 \leq 600$.

De modo a obter melhor resultado em 50 execuções, em menos tempo computacional, foram utilizados os valores dos parâmetros da Tabela 4.2. Em todas as execuções o valor da função objetivo é igual a 0 (Figura 4.6) com um tempo médio de 0,3240s. O valor da função objetivo foi o mesmo encontrado pelo valor analítico. Os valores de x_1, x_2 do valor analítico foi igual a 0, enquanto os valores encontrados pelo método Alcateia para estes valores (Tabela 4.3) são muito próximos aos valores encontrados pelo analítico.

Tabela 4.2: Valores dos parâmetros da função Griewank.

Parâmetro	**Valor**
Número de variáveis	2
Número de Lobos	20
Número de Laços Externos	76
Número de Laços Internos	500
Coeficiente de Independência	0,55
Coeficiente de Contração	0,08

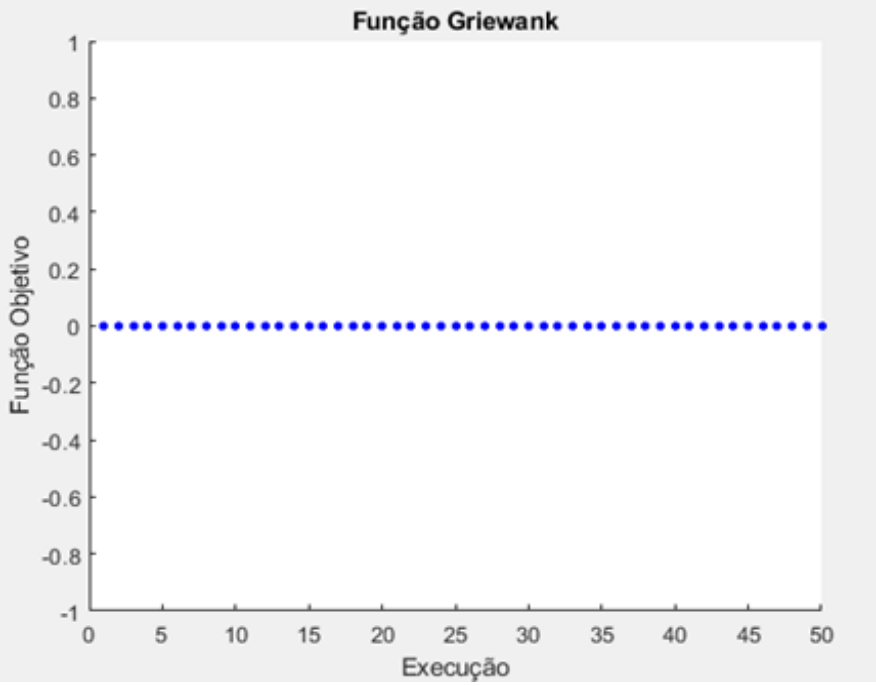

Figura 4.6: Resultados da função Griewank.

Tabela 4.3: Valor de x_1 e x_2 da função Griewank.

Execução	x_1	x_2	**Execução**	x_1	x_2
1	-3,1772e-09	-4,5990e-09	2	-6,5743e-09	1,1802e-08
3	6,0832e-10	-9,9140e-09	4	7,8358e-09	-4,9044e-10
5	-5,9774e-09	1,2895e-08	6	8,3047e-10	1,2478e-08
7	-1,5495e-09	-2,3942e-09	8	1,9459e-09	-1,1814e-08
9	1,8881e-09	-1,1914e-09	10	1,7132e-09	1,4043e-08
11	1,3167e-09	1,2176e-08	12	8,5086e-09	-5,7253e-09
13	9,0545e-09	-3,2503e-09	14	5,2705e-09	1,3237e-08
15	8,6127e-09	1,2138e-08	16	-6,2381e-09	6,1464e-09
17	-5,4913e-09	-6,0056e-09	18	6,5425e-09	3,9313e-09
19	-4,7330e-09	-1,5501e-09	20	2,4554e-09	7,5251e-09
21	-5,9511e-09	6,7332e-09	22	-5,6517e-09	4,1264e-11
23	-1,1930e-09	6,9292e-09	24	-3,0089e-09	5,5528e-09
25	4,5066e-09	-9,2751e-09	26	-3,0952e-09	6,9098e-09
27	-9,0761e-09	-1,3539e-08	28	-2,7726e-10	-1,1748e-08
29	1,0283e-08	2,2987e-09	30	-5,8874e-09	1,2298e-08
31	-9,4163e-09	3,9614e-09	32	8,7238e-09	-1,4468e-08
33	6,5512e-09	-1,1410e-08	34	-7,4125e-09	1,1013e-08
35	-3,8825e-09	1,4864e-08	36	1,0377e-09	1,1288e-08
37	9,3811e-09	-6,8781e-09	38	1,6246e-09	-1,1955e-08
39	9,2162e-09	8,2274e-09	40	2,8174e-09	-1,2552e-08
41	-2,2934e-09	1,1102e-08	42	-6,1778e-09	1,1758e-08

43	5,5510e-09	1,3211e-08	44	9,3221e-09	-1,4279e-08
45	-1,4921e-09	1,4628e-08	46	-7,7370e-09	-2,1224e-09
47	4,5615e-09	1,4689e-08	48	-9,0104e-09	1,1460e-08
49	6,0864e-09	1,0130e-08	50	-3,7034e-09	2,7678e-09

4.1.3 Função Drop-Wave

A função Drop-Wave possui múltiplos mínimos locais e com elevada complexidade. Sua forma é mostrada na Figura 4.7.

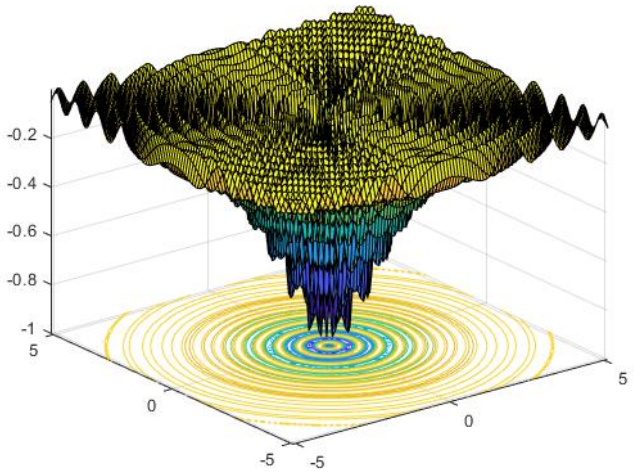

Figura 4.7: Resultado de simulação da função Drop-Wave.

Sua equação foi apresentada por Surjanovic & Bingham (2013):

$$f(x) = -\frac{1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{0{,}5\left(x_1^2+x_2^2\right)+2} \tag{4.3}$$

Seu raio de busca é de $-5{,}12 \leq x_1 \leq 5{,}12$ e $-5{,}12 \leq x_2 \leq 5{,}12$.

Para ter melhor resultado em 50 execuções, em menos tempo computacional, foram utilizados os valores dos parâmetros da Tabela 4.4. Em todas as execuções o valor da função objetivo é o mesmo encontrado pelo valor analítico que é igual a -1 (Figura 4.8), com um tempo médio de 0,037392s. Os valores de x_1 e x_2 do valor

analítico é igual a 0, enquanto os valores encontrados pelo método Alcateia são muito próximo do valor analítico (Tabela 4.5).

Tabela 4.4: Valores dos parâmetros da função Drop-Wave.

Parâmetro	**Valor**
Número de variáveis	2
Número de Lobos	20
Número de Laços Externos	26
Número de Laços Internos	100
Coeficiente de Independência	0,7
Coeficiente de Contração	0,2

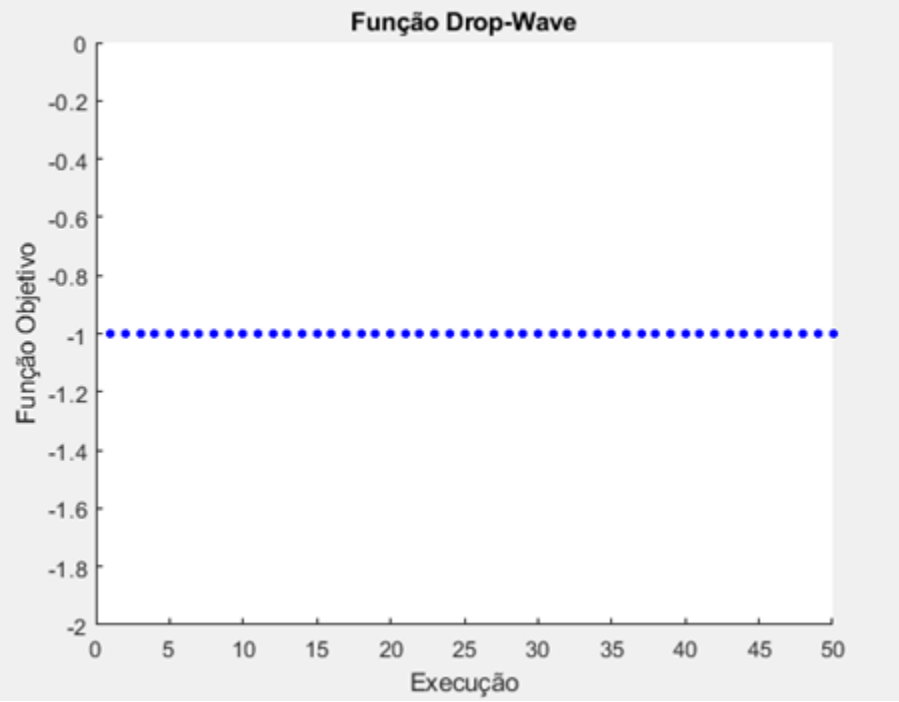

Figura 4.8: Resultados da função Drop-Wave.

Tabela 4.5: Valores de x_1 e x_2 da função Drop-Wave.

Execução	x_1	x_2	**Execução**	x_1	x_2
1	1,1688e-09	-4,8952e-10	2	-1,1056e-09	-7,5439e-10
3	-1,3942e-09	-5,7819e-10	4	-1,7442e-10	8,2022e-10
5	-4,4831e-10	1,2430e-09	6	9,4062e-10	-2,7980e-10

7	-4,1389e-10	-6,6108e-11	8	-4,2123e-10	1,3808e-09
9	1,1823e-09	-3,6402e-10	10	1,6874e-10	9,8933e-10
11	-1,3749e-09	2,8744e-13	12	-1,1594e-09	9,6513e-10
13	-8,3569e-10	1,1032e-09	14	-7,6829e-10	4,5202e-10
15	-9,3931e-10	-9,1960e-10	16	-6,3645e-10	3,5889e-10
17	1,1944e-09	5,3677e-10	18	1,5043e-10	-8,8327e-10
19	2,6456e-10	-3,6817e-10	20	9,2848e-11	-1,4907e-09
21	-4,9156e-10	-6,1381e-10	22	8,7665e-10	1,1720e-09
23	-3,2585e-10	-1,2508e-10	24	-4,9342e-10	-3,5240e-10
25	5,5486e-10	-2,5451e-10	26	-6,0861e-10	-4,5242e-11
27	7,1055e-10	-1,2347e-09	28	5,7458e-10	1,1764e-09
29	-2,5333e-10	-5,1655e-10	30	-9,4579e-10	-7,2465e-10
31	6,8288e-10	1,3138e-09	32	-7,7820e-10	-6,6453e-10
33	-1,1201e-09	1,1581e-10	34	-1,1672e-10	7,2965e-10
35	5,4451e-10	-8,0597e-10	36	1,4030e-09	4,8926e-10
37	-8,4510e-10	-6,9582e-10	38	-8,6976e-10	-1,3088e-10
39	7,0330e-11	-1,2556e-09	40	-8,7849e-10	-1,1982e-09
41	7,1105e-10	8,7346e-10	42	-1,0691e-09	7,0347e-11
43	1,0337e-09	-4,9343e-10	44	-5,6547e-10	-2,9238e-10
45	3,8242e-10	4,7760e-10	46	-1,1491e-09	2,8956e-10
47	-7,2049e-10	3,0994e-10	48	-1,9010e-10	1,9847e-10
49	1,9939e-10	-7,4026e-11	50	-1,2717e-10	-7,7721e-10

Na Figura 4.9 temos o resultado do método quando aumenta o valor do coeficiente de contração de 20% para 30%, melhorando o tempo de execução de 0,037392s para 0,025722s, porém com uma pequena perda no resultado.

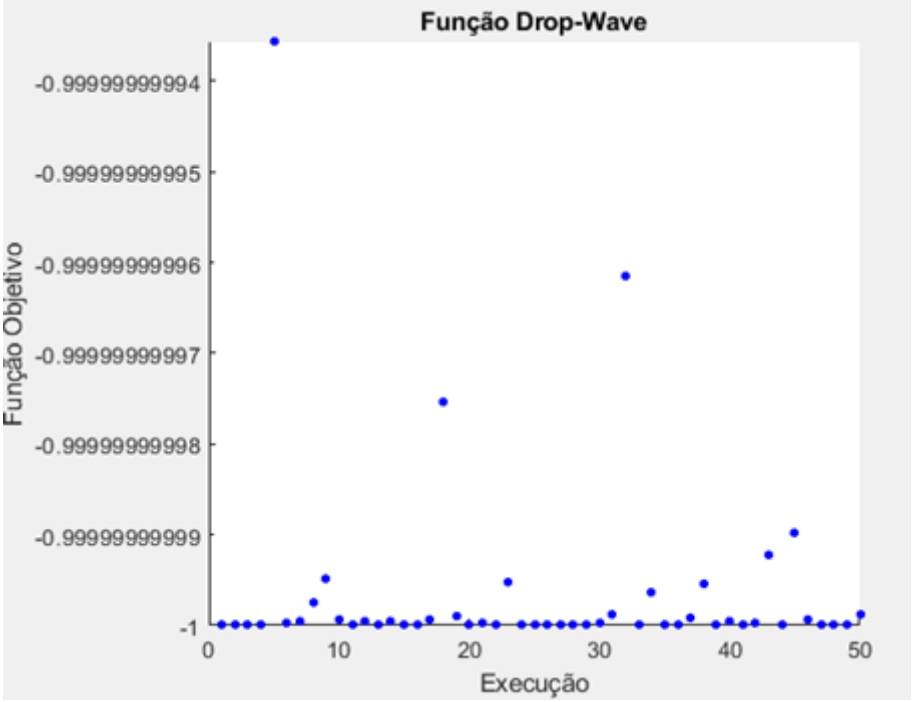

Figura 4.9: Resultado da função Drop-Wave com coef. de contração reduzido.

4.2 Funções Multimodais restritas

As funções multimodais com restrições estudadas nesta pesquisa são: Mishra, Rosenbrock restringida a um disco e a Townsend.

4.2.1 Função Mishra

A função Mishra é uma função multimodal que também é conhecida na literatura como Mishra's Bird function. A referida função encontra-se apresentada na Figura 4.10.

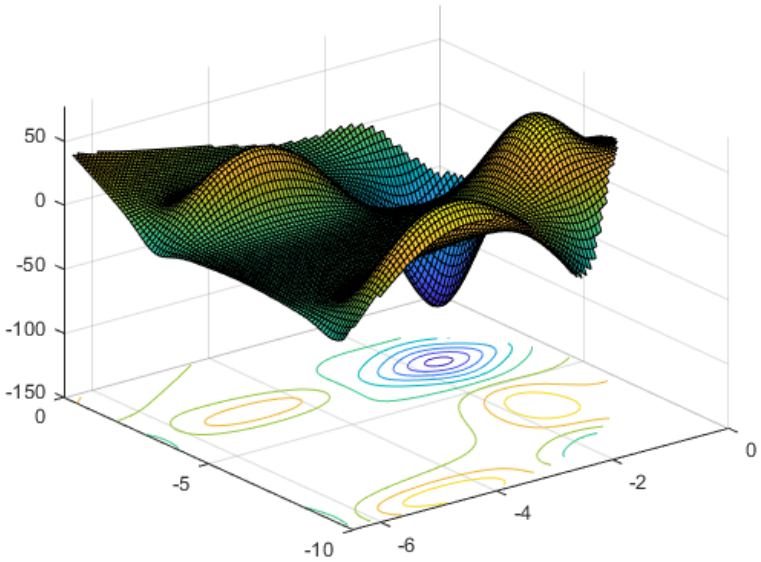

Figura 4.10: Resultado de simulação da função Mishra restrita.

Sua equação é apresentada por Mishra (2006):

$$f(x) = \text{sen}(x_2)\exp\left(1 - \cos(x_1)\right)^2 + \cos(x_1)\exp\left(1 - \text{sen}(x_2)\right)^2 + (x_1 - x_2)^2 \quad (4.4)$$

Com restrição:

$$(x_1 + 5)^2 + (x_2 + 5)^2 < 25 \quad (4.5)$$

E seu raio de busca $-10 \leq x_1 \leq 0$ e $-10 \leq x_2 \leq 0$.

Em todas as 50 execuções o algoritmo Alcateia teve êxito para resolver a Função Mishra em um tempo médio de 0,0356s. Com os valores dos parâmetros da Tabela 4.6 os valores são iguais para a função objetivo (Figura 4.11) e para os valores de x_1 e x_2. em todas as execuções. Os valores de x_1 = -3.1302468, x_2 = -1.5821422 e $f(x)$ = -106.7645367 são os mesmos valores encontrados pelo valor analítico.

Tabela 4.6: Valores dos parâmetros da Função Mishra.

Parâmetro	Valor
Número de variáveis	2
Número de Lobos	13
Número de Laços Externos	45
Número de Laços Internos	100
Coeficiente de Contração	0,1
Coeficiente de Independência	0,8

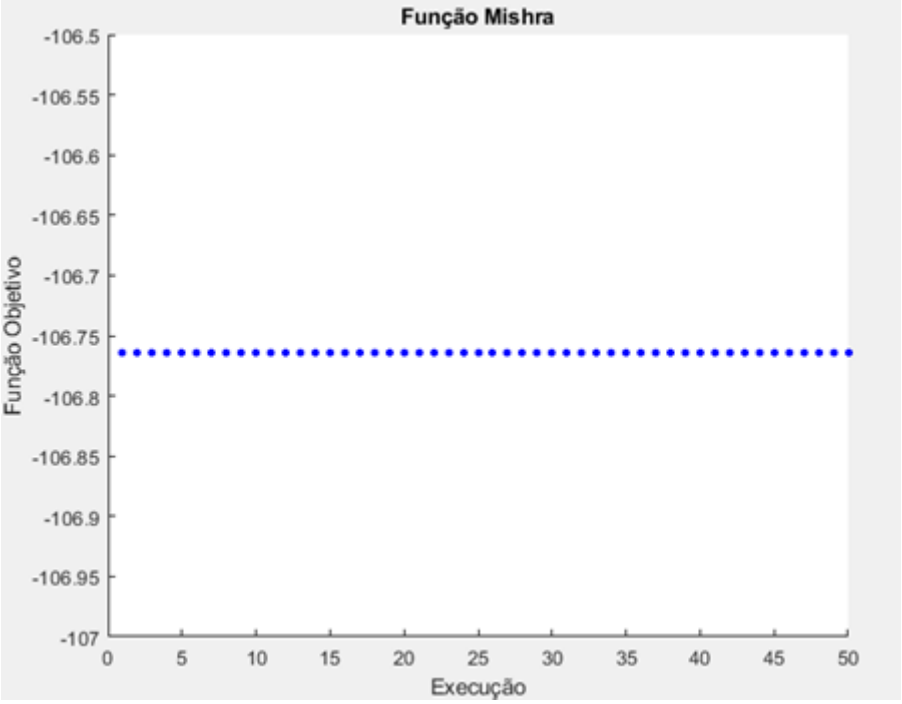

Figura 4.11: Resultados da função Mishra.

4.2.2 Função Townsend

A função Townsend com restrição é proposta por Townsend (2014) e é apresentada na Figura 4.12.

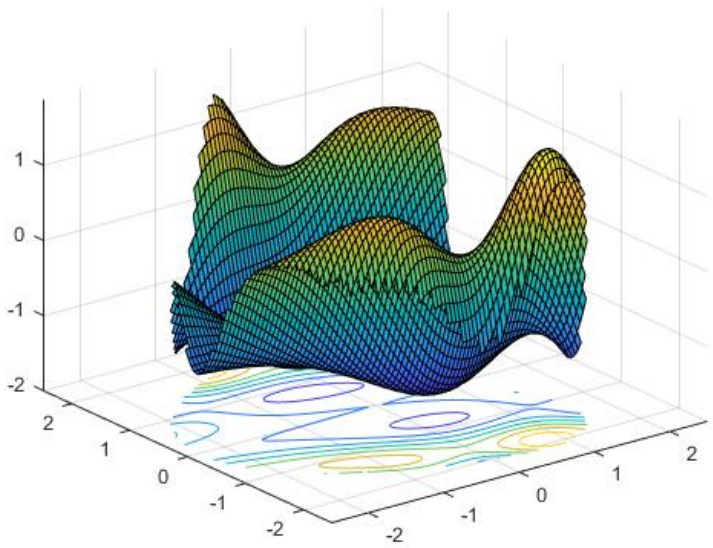

Figura 4.12: Resultado de simulação da função Townsend.

É descrita pela equação:

$$f(x) = -\left(\cos\left((x_1 - 0{,}1)x_2\right)\right)^2 - x_1 \operatorname{sen}(3x_1 + x_2) \tag{4.6}$$

Com restrição:

$$x_1{}^2 + x_2{}^2 < \left(2\cos t - \frac{1}{2}\cos 2t - \frac{1}{4}\cos 3t - \frac{1}{8}\cos 4t\right)^2 + (2\operatorname{sen} t)^2 \tag{4.7}$$

$$\text{Para } t = \operatorname{atan2}(x_1, x_2)$$

Seu raio de busca -2,25 ≤ x_1 ≤ 2,5 e -2,25 ≤ x_2 ≤ 2,5.

Nas 50 execuções foram utilizados os valores dos parâmetros da Tabela 4.7. Em um tempo médio de 0,1473s, o maior valor que o método Alcateia encontra para a função objetivo é -2,0239807 e o menor é -2,0239883 (Figura 4.13), valores muito próximos do valor analítico que é -2,0239884. Os valores analíticos de x_1 e x_2 são 2,0052938 e 1,1944509 respectivamente e valores x_1 e x_2 do algoritmo Alcateia são valores muito próximos dos valore analíticos (Tabela 4.8).

Tabela 4.7: Valores dos parâmetros da Função Townsend.

Parâmetro	Valor
Número de variáveis	2
Número de Lobos	15
Número de Laços Externos	20
Número de Laços Internos	1000
Coeficiente de Independência	0,4
Coeficiente de Contração	0,3
Tempo de Execução	0,1473s

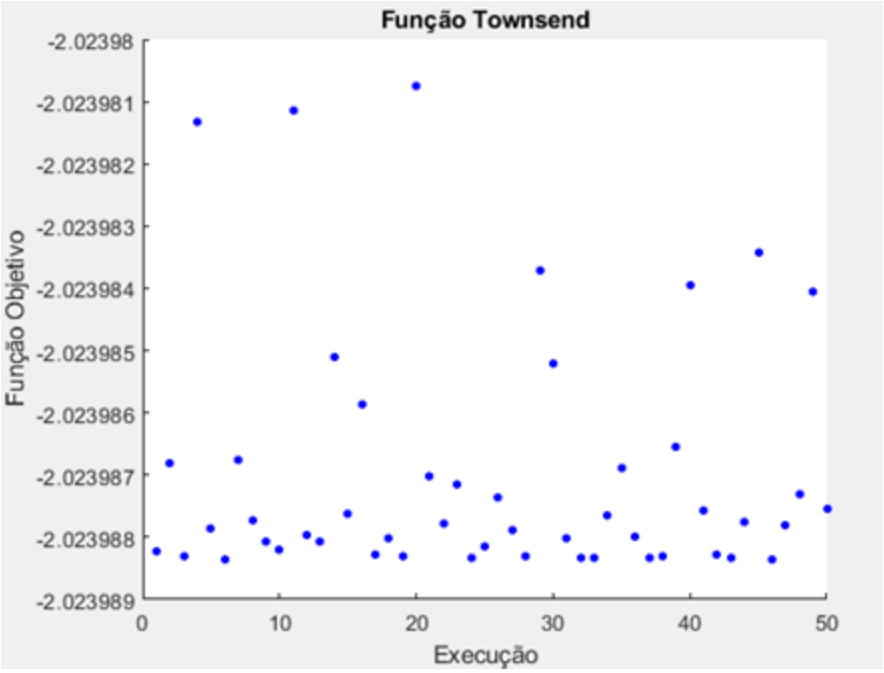

Figura 4.13: Resultado da função Townsend.

Tabela 4.8: Valores de x_1, x_2 e $f(x)$.

Execução	x_1	x_2	$f(x)$
1	2,0052075	1,1946073	-2,0239882
2	2,0049882	1,1950043	-2,0239868
3	2,0052315	1,1945638	-2,0239883
4	2,0059415	1,1932750	-2,0239813

5	2,0051200	1,1947657	-2,0239879
6	2,0052735	1,1944876	-2,0239884
7	2,0056011	1,1938936	-2,0239868
8	2,0051072	1,1947890	-2,0239877
9	2,0051627	1,1946885	-2,0239881
10	2,0053934	1,1942704	-2,0239882
11	2,0046395	1,1956345	-2,0239811
12	2,0054490	1,1941695	-2,023988
13	2,0051580	1,1946970	-2,0239881
14	2,0048502	1,1952538	-2,0239851
15	2,0055048	1,1940683	-2,0239876
16	2,0049059	1,1951531	-2,0239859
17	2,0053604	1,1943303	-2,0239883
18	2,0051505	1,1947105	-2,023988
19	2,0052395	1,1945493	-2,0239883
20	2,0046167	1,1956757	-2,0239807
21	2,0050088	1,1949670	-2,023987
22	2,0051090	1,1947857	-2,0239878
23	2,0055626	1,1939634	-2,0239871
24	2,0053338	1,1943785	-2,0239883
25	2,0054043	1,1942506	-2,0239882
26	2,0050467	1,1948984	-2,0239873
27	2,0054636	1,1941430	-2,0239879
28	2,0053547	1,1943405	-2,0239883
29	2,0047659	1,1954061	-2,0239837
30	2,0048584	1,1952390	-2,0239852
31	2,0051477	1,1947156	-2,023988
32	2,0053388	1,1943694	-2,0239883
33	2,0052428	1,1945433	-2,0239883
34	2,0050888	1,1948222	-2,0239877
35	2,0055885	1,1939165	-2,0239869
36	2,0054397	1,1941863	-2,023988
37	2,0052584	1,1945150	-2,0239883
38	2,0053544	1,1943410	-2,0239883
39	2,0056223	1,1938551	-2,0239865
40	2,0058061	1,1935213	-2,0239839
41	2,0055118	1,1940556	-2,0239876
42	2,0053587	1,1943333	-2,0239883
43	2,0053272	1,1943904	-2,0239883
44	2,0051034	1,1947959	-2,0239878
45	2,0047485	1,1954376	-2,0239834
46	2,0053147	1,1944130	-2,0239884
47	2,0051118	1,1947806	-2,0239878

48	2,0055426	1,1939997	-2,0239873
49	2,0047850	1,1953717	-2,0239841
50	2,0055139	1,1940518	-2,0239875

4.2.3 Função Rosenbrock restringida a um disco

A função Rosenbrock foi proposta por Rosenbrock (1960) e é dada pela equação:

$$f(x) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i{}^2)^2 + (x_i - 1)^2 \quad (4.8)$$

Com restrição (SOLVE, 2020):

$$x_i{}^2 + x_{i+1}{}^2 \leq 2 \quad (4.9)$$

Na forma bidimensional, seu gráfico é representado pela Figura 4.14 e seu raio de busca é de -1,5 ≤ x_1 ≤ 1,5 e -1,5 ≤ x_2 ≤ 1,5.

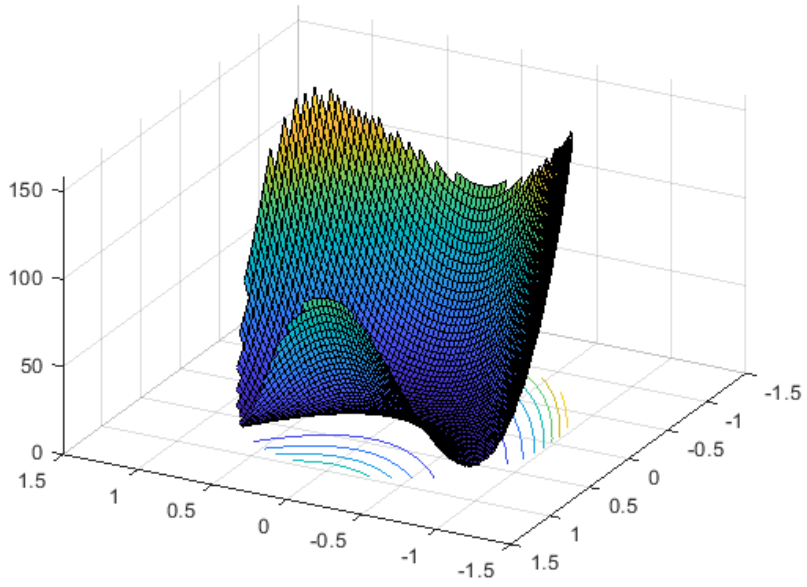

Figura 4.14: Resultado de simulação da função Rosenbrock.

Foram apresentados, como valores, os parâmetros apresentados na Tabela 4.9. O algoritmo Alcateia encontrou x_1 = 0,7864, x_2 = 0.6176 e $f(x)$ = 0,04567 em todas

as 50 execuções (Figura 4.15), com um tempo médio de 0,0153s. O algoritmo teve sucesso, pois estes valores foram os mesmos encontrados pelos valores analíticos.

Tabela 4.9: Valores de parâmetros da função Rosenbrock.

Parâmetro	Valor
Número de variáveis	2
Número de Lobos	8
Número de Laços Externos	42
Número de Laços Internos	80
Coeficiente de Contração	0,1
Coeficiente de Independência	0,8

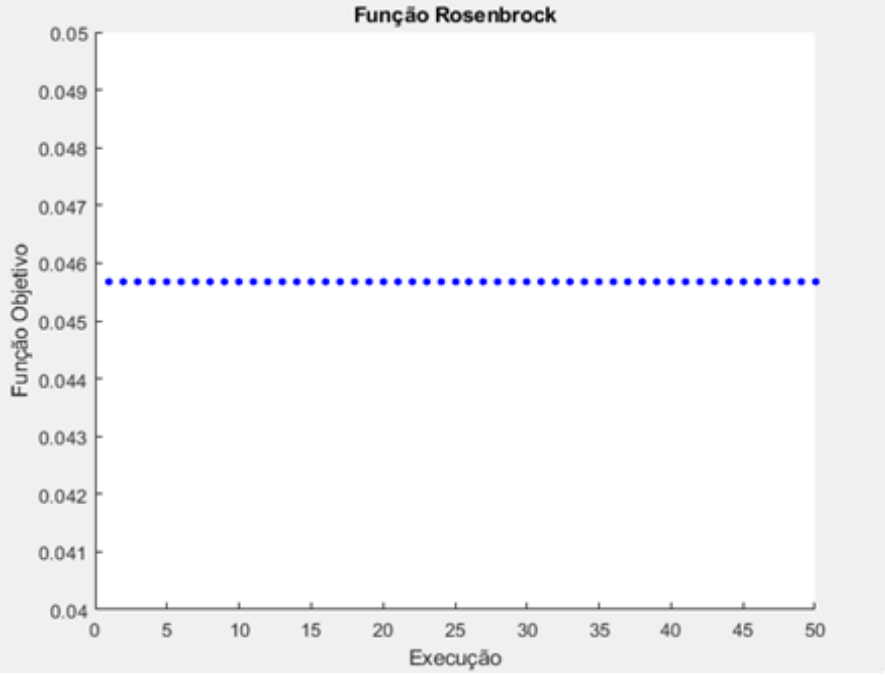

Figura 4.15: Resultado da função Rosenbrock.

4.3 Projetos de Engenharia

Os projetos de Engenharia estudados na presente pesquisa foram: o projeto da viga de aço presente em Rao (2009), o projeto de vaso de pressão que segue a

formulação de Kannan e Karmer (1994), e o projeto de redutor de velocidade (caixa de transmissão) presente em Rao (2009). Estes problemas já foram abordados por diferentes literaturas que, no presente trabalho, tiveram seus resultados comparados aos do Alcateia. Abaixo uma breve descrição destes projetos.

4.3.1 Projeto de uma viga de aço

O problema do projeto da viga de aço tem como objetivo minimizar o custo de fabricação de uma viga confeccionada em aço e que está sujeita a algumas restrições de projeto: Tensão cisalhante admissível, tensão de flexão admissível e dimensões. Para levar em conta todas essas restrições, são atribuídas ao problema, quatro variáveis, a saber: espessura da solda (h, x_1); comprimento da junta de solda (l, x_2); largura do feixe (t, x_3); espessura da viga (b, x_4). A Figura 4.16 ilustra geometricamente o problema, como proposto por Rao (2009).

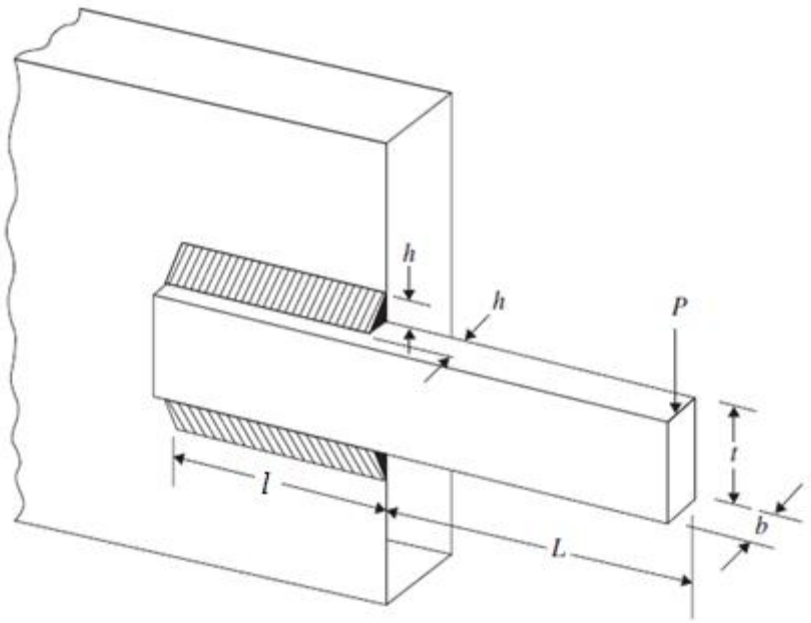

Figura 4.16: Ilustração da viga soldada.
Fonte: Alfares e Esat (2006).

Equação a ser minimizada:

$$\mathrm{f}(\mathrm{x}) = 1{,}10471\,\mathrm{x}_1^2\,\mathrm{x}_2 + 0{,}04811\,\mathrm{x}_3\,\mathrm{x}_4(14 + \mathrm{x}_2) \tag{4.10}$$

Sujeita às restrições:

Onde:

$$\tau - \tau_{adm} \leq 0 \tag{4.11}$$

$$\sigma - \sigma_{adm} \leq 0 \tag{4.12}$$

$$h - b \leq 0 \tag{4.13}$$

$$0{,}10471h^2 + 0{,}04811tb(14 + l) - 5 \leq 0 \tag{4.14}$$

$$0{,}125 - h \leq 0 \tag{4.15}$$

$$\delta - \delta_{adm} \leq 0 \tag{4.16}$$

$$P - P_c \leq 0 \tag{4.17}$$

$$\tau = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{1}{2R} + (\tau'')^2} \tag{4.18}$$

$$\tau' = \frac{P}{\sqrt{2}hl} \tag{4.19}$$

$$\tau'' = \frac{MR}{J} \tag{4.20}$$

$$M = P(L + 0{,}5) \tag{4.21}$$

$$R = \sqrt{\frac{t^2}{4} + \left(\frac{h+t}{2}\right)^2} \tag{4.22}$$

$$J = 2\left\{\sqrt{2}hl\left[\frac{t^2}{12} + \left(\frac{h+t}{2}\right)^2\right]\right\} \tag{4.23}$$

$$\sigma = \frac{6PL}{bt^2} \tag{4.24}$$

$$\delta = \frac{4PL^3}{Eb^3} \tag{4.25}$$

$$P_c = \frac{4{,}013E\sqrt{\frac{t^2b^6}{36}}}{L^2} + \left(1 - \frac{t}{2L}\sqrt{\frac{E}{4G}}\right) \tag{4.26}$$

Sendo ainda:

$P = 6000$; $L = 14$; $E = 3 \times 10^7$; $G = 1{,}2 \times 10^7$; $\tau_{adm} = 13600$; $\sigma_{adm} = 3 \times 10^4$; $\delta_{max} = 0{,}25$.

E os intervalos de busca sendo: $0{,}1 \leq h \leq 2$; $0{,}1 \leq l \leq 10$; $0{,}1 \leq t \leq 10$; $0{,}1 \leq b \leq 2$.

4.3.1.1 Resultados do projeto da viga de aço

As abordagens aplicadas a esse problema incluem programação geométrica (RAGSDELL; PHILLIPS, 1976), algoritmo genético com representação binária e função de penalidade tradicional (DEB, 1991), algoritmos genéticos através do uso de seleção de torneios baseados em dominância (COELLO; MONTES, 2002), algoritmo de otimização efetiva de enxame de partículas coevolucionárias (HE; WANG, 2007), algoritmo de otimização de enxame de partículas coevolutivo híbrido (ZHOU; PEI, 2010) e algoritmo Alcateia.

Em 50 execuções, o algoritmo Alcateia obteve uma consistência em seu valor da função objetivo, como pode ser visto na Figura 4.17. Usando os valores de parâmetros da Tabela 4.10, tem-se um tempo média de execução de 1,21002s.

Em comparação aos outros algoritmos, o algoritmo Alcateia é o que tem o melhor resultado em relação ao valor da função objetivo (Tabela 4.11). O algoritmo Alcateia encontrou o valor ótimo igual a 1,578916 e o segundo melhor é o algoritmo de enxame de partículas coevolutivo híbrido (ZHOU; PEI, 2010) que tem o valor igual a 1,724852, uma diferença de 8.46%.

Tabela 4.10: Valores dos parâmetros da Viga de Aço do Alcateia.

Parâmetro	**Valor**
Número de variáveis	4
Número de Lobos	35
Número de Laços Externos	45
Número de Laços Internos	200
Coeficiente de Independência	0,7
Coeficiente de Contração	0

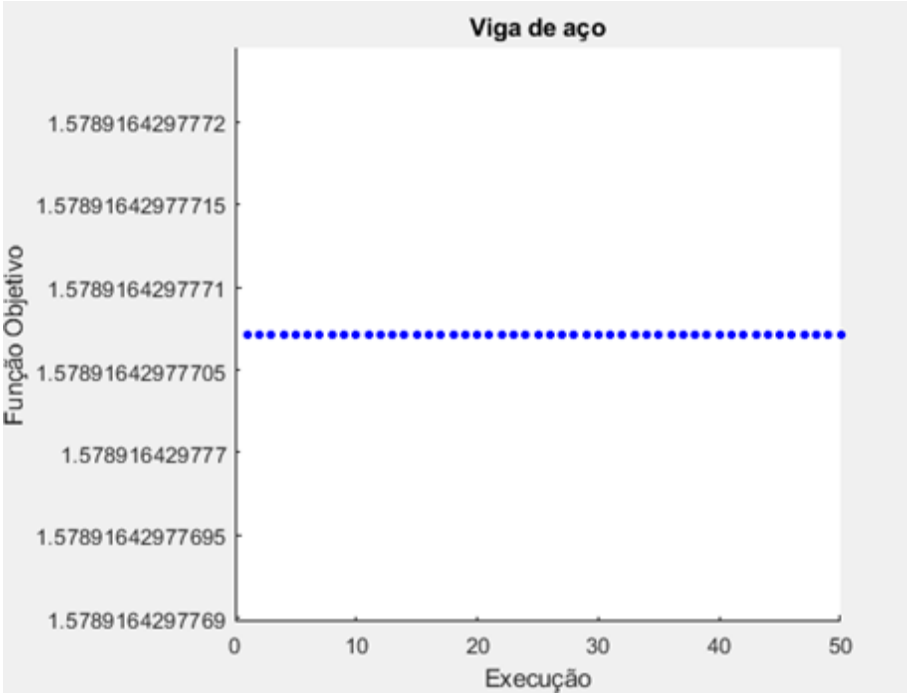

Figura 4.17: Resultado da viga de aço.

Tabela 4.11: Comparação dos resultados do projeto de viga de aço.

Variáveis	Alcateia	Ragsdell e Phillips (1976)	Deb (1991)	Coelho e Montes (2002)	He e Wang (2007)	Zhou e Pei (2010)
x_1	0,182854	0,245500	0,248900	0,205986	0,202369	0,205729
x_2	3,285575	6,196000	6,173000	3,471328	3,544214	3,470489
x_3	9,585229	8,273000	8,178900	9,020224	9,048210	9,036624
x_4	0,182854	0,245500	0,253300	0,206480	0,205723	0,205729

f(x)	1,578916	2,385937	2,433116	1,728226	1,728024	1,724852

4.3.2 Projeto do redutor de velocidade.

Esse problema propõe a minimização do custo com material de um redutor de velocidade, onde as dimensões do equipamento são restringidas pela tensão entre os dentes da engrenagem, tensão superficial e tensões nos eixos. As variáveis a serem consideradas são: Largura da face x_1, módulo x_2, número de dentes do pinhão x_3, comprimento dos eixos x_4 e x_5, e os diâmetros dos eixos x_6 e x_7. A Figura 4.18 mostra geometricamente o problema, como proposto por (RAO, 2009).

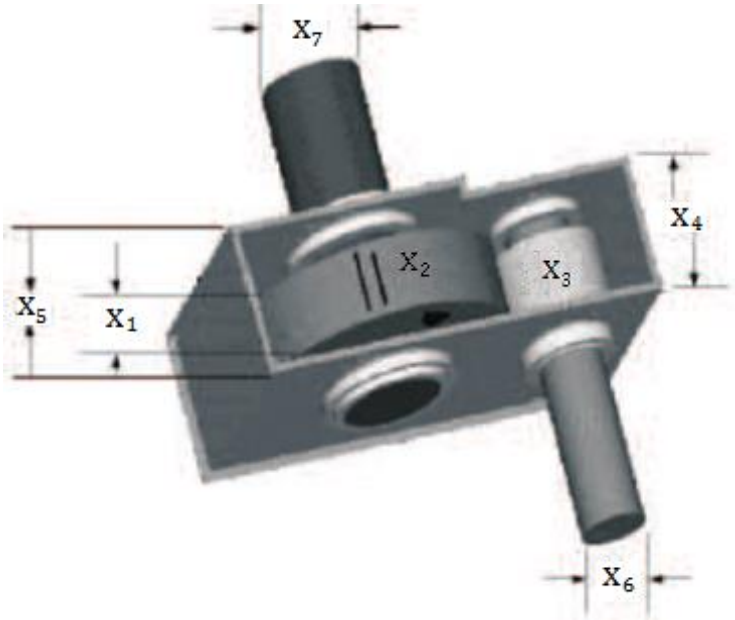

Figura 4.18: Projeto de redutor de velocidade.
Fonte: Brajevic (2010).

A equação a ser minimizada:

$$f(x) = 0{,}7854x_1{x_2}^2(3{,}3333{x_3}^2 + 14{,}9334x_3 - 43{,}0934)$$
$$-1{,}508x_1({x_6}^2 + {x_7}^2) + 7{,}4777({x_6}^3 + {x_7}^3) + 0{,}78054(x_4{x_6}^2 + x_5{x_7}^2) \qquad (4.27)$$

Restringida por:

$$\frac{27}{x_1 x_2^{\,2} x_3^{\,2}} - 1 \leq 0 \tag{4.28}$$

$$\frac{397{,}5}{x_1 x_2^{\,2} x_3^{\,2}} - 1 \leq 0 \tag{4.29}$$

$$\frac{1{,}93 x_4^{\,3}}{x_2 x_3 x_6^{\,4}} - 1 \leq 0 \tag{4.30}$$

$$\frac{1{,}93 x_5^{\,3}}{x_2 x_3 x_7^{\,4}} - 1 \leq 0 \tag{4.31}$$

$$\frac{1}{10 x_6^{\,3}} \sqrt{\left(\frac{745 x_4}{x_2 x_3}\right)^2 + 16{,}9 \times 10^6} - 1100 \leq 0 \tag{4.32}$$

$$\frac{1}{10 x_7^{\,3}} \sqrt{\left(\frac{750 x_5}{x_2 x_3}\right)^2 + 157{,}5 \times 10^6} - 850 \leq 0 \tag{4.33}$$

$$\frac{1{,}5 x_6 + 1{,}9}{x_4} - 1 \leq 0 \tag{4.34}$$

$$\frac{1{,}1 x_7 + 1{,}9}{x_5} - 1 \leq 0 \tag{4.35}$$

$$x_2 x_3 - 40 \leq 0 \tag{4.36}$$

$$5 - \frac{x_2}{x_1} \leq 0 \tag{4.37}$$

$$\frac{x_1}{x_2} - 12 \leq 0 \tag{4.38}$$

Além das restrições, foi considerado o espaço de busca:

$2{,}6 \leq x_1 \leq 3{,}6$; $0{,}7 \leq x_2 \leq 0{,}8$; $17 \leq x_3 \leq 28$; $7{,}3 \leq x_4 \leq 8{,}3$; $7{,}3 \leq x_5 \leq 8{,}3$; $2{,}9 \leq x_6 \leq 5{,}0$; $5 \leq x_7 \leq 5{,}5$.

4.3.2.1 Resultado do projeto do redutor de velocidade

As abordagens aplicadas são algoritmo de otimização de enxame de partículas (CAGNINA, 2008), algoritmo de colônia de abelhas artificial aprimorado para problemas restritos (BRAJEVIC,2010) e algoritmo Alcateia.

Nas 50 execuções o algoritmo Alcateia precisa de um tempo médio de 1,589375s para obter o mesmo valor para função objetivo em todas as execuções (Figura 4.19).

Tendo como valor dos parâmetros a Tabela 4.12, o Alcateia encontrou para a função objetivo o valor de 2994,513287. Em relação aos outros algoritmos, que alcançaram o mesmo resultado igual a 2996,348165 para função objetivo, o Alcateia tem um melhor resultado, apesar de ser uma pequena diferença de 0,061% (Tabela 4.13).

Tabela 4.12: Valores dos parâmetros do redutor de velocidade do Alcateia.

Parâmetro	**Valor**
Número de variáveis	7
Número de Lobos	35
Número de Laços Externos	67
Número de Laços Internos	1000
Coeficiente de Independência	0,3
Coeficiente de Contração	0,1

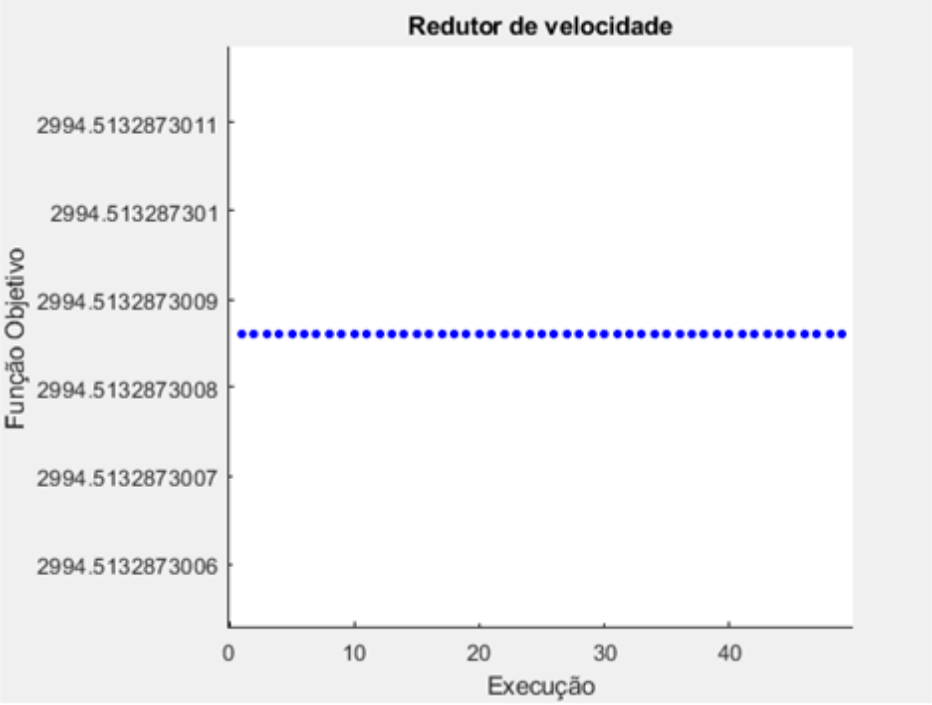

Figura 4.19: Resultado do redutor de velocidade.

Tabela 4.13: Comparação dos resultados do projeto de redutor de velocidade.

Variáveis	Alcateia	Cagnina (2008)	Brajevic (2010)
x_1	3,500000	3,500000	3,500000
x_2	0,700000	0,700000	0,700000
x_3	17,00000	17,000000	17,000000
x_4	7,300000	7,300000	7,300000
x_5	7,715320	7,800000	7,800000
x_6	3,350380	3,350215	3,350214
x_7	5,286654	5,286683	5,286683
$f(x)$	2994,513287	2996,348165	2996,348165

4.3.3 Projeto de um vaso de pressão

Esse último problema a ser analisado procura minimizar o custo com material, soldagem e conformação mecânica de um vaso de pressão cilíndrico. As variáveis desse projeto apenas levam em consideração parâmetros dimensionais, a saber: A

espessura da casca (T_s, x_1); a espessura da tampa (T_h, x_2); o raio interno (R, x_3); o comprimento da secção cilíndrica (L, x_4).

Devido aos padrões de chapas de aço, os valores para T_s e T_h são limitados a múltiplos de 0.065". O projeto de vaso de pressão que segue a formulação de Kannan e Karmer (1994), é mostrado na Figura 4.20.

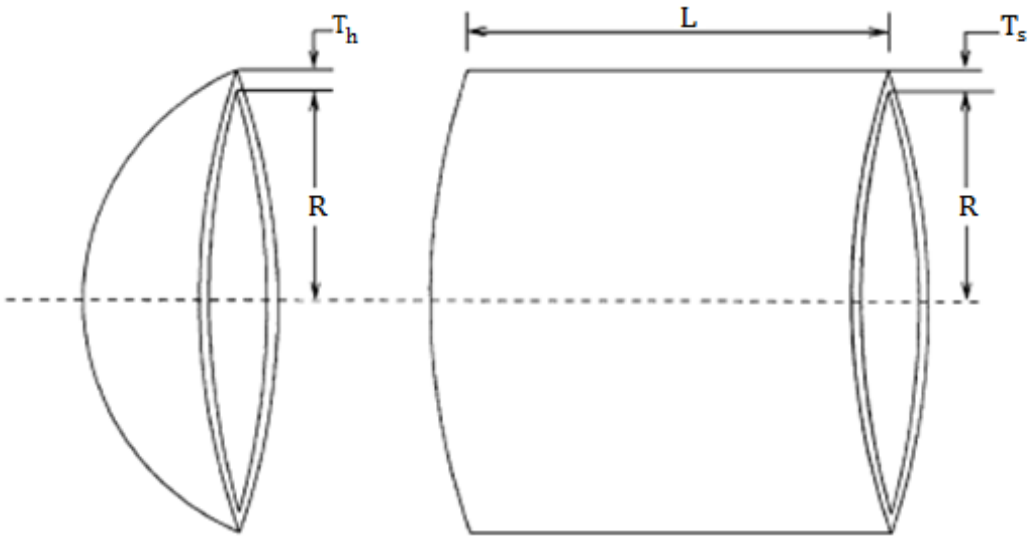

Figura 4.20: Ilustração do vaso de pressão e suas dimensões.
Fonte: He e Wang (2007).

A equação a ser minimizada é:

$$f(x) = 0{,}6224x_1x_3x_4 + 1{,}7781x_2{}^2x_3 + 3{,}1661x_1{}^2x_4 + 19{,}84x_1{}^2x_3 \qquad (4.39)$$

Sendo esta, restringida por:

$$-x_1 + 0{,}0193x_3 \leq 0 \qquad (4.40)$$

$$-x_2 + 0{,}00954x_3 \leq 0 \qquad (4.41)$$

$$-\pi x_3{}^2x_4 - 0{,}75\pi x_3{}^3 + 12{,}96 \times 10^5 \leq 0 \qquad (4.42)$$

$$x_4 - 240 \leq 0 \qquad (4.43)$$

Ainda foi configura um espaço de busca definido por: $0{,}0625 \leq x_1 \leq 6{,}1875$;

$0{,}0625 \leq x_2 \leq 6{,}1875$; $10 \leq x_3 \leq 200$; $10 \leq x_4 \leq 200$.

4.3.3.1 Resultados do projeto de um vaso de pressão

As abordagens aplicadas são método baseado em multiplicador de intervalo aumentado (KANNAN; KRAMER,1994), Algoritmos evolutivos em aplicações de engenharia (DEB, 1997), algoritmos genéticos através do uso de seleção de torneios baseados em dominância(COELLO; MONTES, 2002), algoritmo de otimização efetiva de enxame de partículas coevolucionárias (HE; WANG, 2007), algoritmo de otimização de enxame de partículas coevolutivo híbrido (ZHOU; PEI, 2010) e do Alcateia.

Devido a sua maior complexidade em relação aos outros métodos de engenharia ora citados, o Alcateia precisa de um tempo médio de execução de 23,746416s. Tempo de execução bem maior que o Alcateia precisa para solucionar o projeto da viga de aço que é de 1,210021s e o projeto do redutor de velocidade que é de 1,589375s.

Nas 50 execuções o Alcateia também tem o mesmo resultado em todas as execuções (Figura 4.21), usando os valores dos parâmetros da Tabela 4.14. O valor da função objetivo é igual a 5885,3328, melhor resultado em relação aos demais (Tabela 4.15). Comparando com o segundo melhor, algoritmo de otimização de enxame de partículas coevolutivo híbrido (ZHOU; PEI, 2010), que tem o valor da função objetivo igual a 6059,7143, tem-se uma diferença de 2,88%.

Tabela 4.14: Valores dos parâmetros do Alcateia no projeto vaso de pressão.

Parâmetro	**Valor**
Número de variáveis	4
Número de Lobos	70
Número de Laços Externos	100
Número de Laços Internos	1000
Coeficiente de Contração	0,005
Coeficiente de Independência	0,4

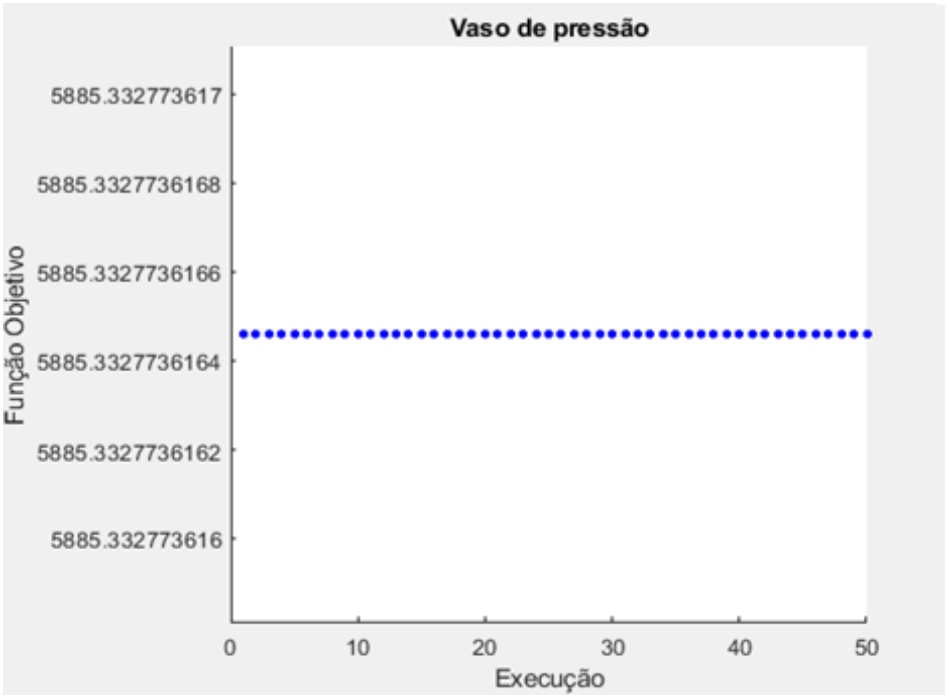

Figura 4.21: Resultado do vaso de pressão.

Tabela 4.15: Comparação dos resultados do projeto de vaso de pressão.

Variá-veis	Alcateia	Kannan e Kramer (1994)	Deb (1997)	Coelho e Montes (2002)	He e Wang (2007)	Zhou e Pei (2010)
x_1	0,778169	1,125000	0,937500	0,812500	0,812500	0,812500
x_2	0,384649	58,291000	0,5000000	0,437500	0,437500	0,437500
x_3	40,319619	43,690000	48,329000	42,097398	42,091266	42,098446
x_4	200,00000	0,000016	112,67900	176,65405	176,74650	176,63659
$f(x)$	5885,3328	7198,0428	6410,3811	6059,9463	6061,0777	6059,7143

4.4 Identificação de danos em uma viga

O modelo matemático utilizado para identificação de danos baseia-se na matriz de flexibilidade. O modelo é apresentado em Corrêa (2013).

4.4.1 Modelagem do problema de identificação de danos

A matriz de flexibilidade G de uma estrutura com n graus de liberdade (GDL) é a matriz n×n definida como a inversa de sua matriz de rigidez K. Portanto, em um problema estático, a matriz de flexibilidade relaciona a força f aplicada na estrutura com o deslocamento u resultante,

$$\mathrm{u} = \mathrm{K}^{-1}\mathrm{f} = \mathrm{Gf}. \tag{4.44}$$

A modelagem do problema é realizada utilizando a teoria de autovalores e autovetores para obter as frequências naturais e as formas modais não-amortecidas da estrutura, tem-se

$$(\mathrm{K} - \omega_i{}^2\,\mathrm{M})\,\emptyset_i = 0, \tag{4.45}$$

onde M é a matriz de massa e K a matriz de rigidez, ambas com dimensão n×n. Os termos ω_i e $\emptyset_i$ referem-se, respectivamente, à i-ésima frequência natural e à i-ésima forma modal da estrutura.

De forma geral,

$$\mathrm{K}\Phi = \mathrm{M}\Phi\Lambda, \tag{4.46}$$

onde Φ é a matriz modal da estrutura, de dimensão n×n e Λ é uma matriz diagonal, n×n, formada pelos valores quadráticos das frequências naturais, $\lambda_{ii} = \omega_i{}^2$.

Para as formas modais da estrutura normalizadas em relação à matriz de massa, tem-se

$$\Phi^{\mathrm{T}}\mathrm{M}\Phi = \mathrm{I}; \tag{4.47}$$

$$\Phi^{\mathrm{T}}\mathrm{K}\Phi = \Lambda, \tag{4.48}$$

onde I e a matriz identidade e T representa a transposição de uma matriz.

Considerando a Equação (4.48) e o fato da matriz de flexibilidade de uma estrutura ser a inversa da matriz de rigidez, tem-se

$$\mathrm{G} = (\Phi\Lambda^{-1}\Phi^{\mathrm{T}}) = \sum_{\mathrm{i}=1}^{\mathrm{n}} \frac{1}{\omega_{\mathrm{i}}{}^{2}} \Phi_{\mathrm{i}}\Phi_{\mathrm{i}}{}^{\mathrm{T}}\,. \tag{4.49}$$

Devido às limitações experimentais, na prática tem-se a seguinte aproximação para a matriz de flexibilidade experimental $\mathrm{G}_{\mathrm{exp}}$ da estrutura,

$$\mathrm{G}_{\mathrm{exp}} = \sum_{\mathrm{i}=1}^{\mathrm{n}_{\mathrm{exp}}} \frac{1}{\omega_{\mathrm{i,exp}}^{2}} \Phi_{\mathrm{i,exp}} \otimes \Phi_{\mathrm{i,exp}}\,, \tag{4.50}$$

com $\mathrm{n}_{\mathrm{exp}} < \mathrm{n}$ o número de modos obtidos do ensaio experimental, $\omega_{\mathrm{i,exp}}$ e $\Phi_{\mathrm{i,exp}}$ são a i-ésima frequência natural não-amortecida e forma modal obtidas experimentalmente, respectivamente.

Uma boa estimativa para a matriz de flexibilidade pode ser obtida experimentalmente a partir de modos de mais baixa frequência da estrutura. Para se definir um problema de identificação de danos estruturais baseado na matriz de flexibilidade, torna-se necessária a determinação de uma matriz de flexibilidade analítica reduzida $\bar{\mathrm{G}}$ relacionada apenas aos m GDL medidos no ensaio de vibrações que contenha informações a respeito das propriedades de rigidez da estrutura como um todo. Para tal, a matriz de rigidez original deve ser particionada na forma

$$K = \begin{bmatrix} K_{mm} & K_{mo} \\ K^{T}_{mo} & K_{oo} \end{bmatrix}, \tag{4.51}$$

onde os índices m e *o* referem-se, respectivamente, aos GDL medidos e omitidos. Alvim, Peterson e Park (1995) mostram que $\bar{G}$ é igual a inversa da matriz rigidez reduzida, obtida pela redução estática de Guyan (1965), então

$$\bar{G} = [K_{mm} - K_{mo} K^{-1}_{oo} K^{T}_{mo}]^{-1}. \tag{4.52}$$

O problema de identificação de danos em análise visa a minimização, em relação ao parâmetro de coesão β, de um erro baseado na diferença entre a matriz de flexibilidade experimental e a matriz de flexibilidade analítica obtido através do Método de Elementos Finitos (MEF).

Definindo-se o vetor de parâmetros de coesão $\beta = [\beta_1, \beta_2, \ldots, \beta_{np}]$, onde np é o número total de parâmetros de coesão do modelo, G_{exp} e $\bar{G}(\beta)$ a matriz de flexibilidade experimental e analítica, respectivamente. O problema de identificação de danos pode ser definido como um problema de otimização onde se deseja minimizar o funcional,

$$\mathcal{F}(\beta) = \frac{\|G_{ex} - \bar{G}(\beta)\|^2}{2\|G_{ex} - \bar{G}_0\|^2}, \text{ satisfazendo } 0 \leq \beta_j \leq 1,\ j = 1, \ldots, np, \tag{4.53}$$

onde $\bar{G}_0$ é a matriz de flexibilidade reduzida obtida com a estimativa inicial do vetor β, ou seja, $\bar{G}_0 = \bar{G}(\beta_0)$ e $\|.\|$ refere-se à norma de Frobenius.

4.4.2 Resultados

Considerou-se uma viga de Euler-Bernoulli simplesmente apoiada (Figura 4.22a), de alumínio com 1,000m de comprimento, 0,005m de espessura, 0,050m de

largura, módulo de elasticidade nominal $E_0 = 7{,}2582\times10^{10}Pa$ e momento de inércia de área nominal $I_0 = 5{,}2083\times10^{-10}m^4$. A discretizacão da viga foi realizada pelo Método dos Elementos Finitos (MEF) em 20 elementos, a discretizacão do campo de danos também foi realizada via MEF. Foram adotados elementos com dois nós, onde cada ponto nodal possui dois graus de liberdade (GDL): um transversal e outro de rotação, e um parâmetro de coesão. Logo a estrutura possui 21 nós (Figura 4.22b), 40 GDL, devido as condições de contorno abordadas, e 21 parâmetros nodais de coesão. No entanto, foram medidos apenas 10 GDL transversais e igualmente espaçados.

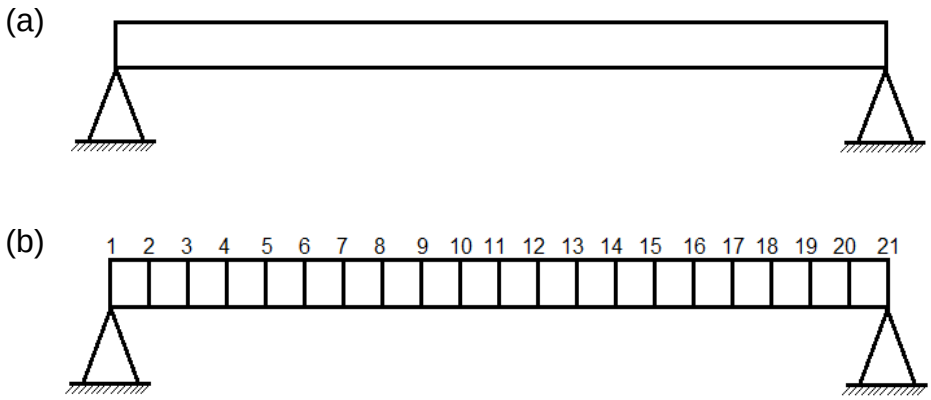

Figura 4.22: (a) Viga apoiada, (b) discretizada em 20 elementos com 21 nós.

Admitiu-se os modos de vibração na faixa de frequência entre 0 − 450Hz. A imposição do defeito à viga é realizada através de uma redução na altura relativa da seção transversal $h(x)/h_0$, sendo h_0 a espessura original da viga e $h(x)$ a espessura na posição danificada. Portanto, nos nós defeituosos tem-se $h(x)/h_0 < 1$, e nos nós onde não há danos tem-se $h(x)/h_0 = 1$.

A imposição de dano é realizada escolhendo-se uma posição e um valor para o parâmetro de coesão, dado pela Equação (4.54)

$$\beta = \left(\frac{h(x)}{h_0}\right)^3, \tag{4.54}$$

onde β varia no intervalo [0,1]. Se $\beta = 1$ a estrutura está sem danos, sendo $\beta = 0$ uma ruptura local deve ser considerada. As posições escolhidas para a imposição do dano são baseadas no trabalho de Corrêa (2013), onde mostra-se que as regiões próximas

aos extremos da viga apresentam menor sensibilidade à presença de dano, sendo assim, mais difícil a correta detecção de danos estruturais.

A porcentagem do dano é calculada através da Equação (4.55)

$$\text{Porcentagem} = 1 - \left(\frac{h(x)}{h_0}\right). \tag{4.55}$$

Analisou-se as posições da viga de 0,20m, 0,25m, 0,40m, 0,45m, 0,85m e 0,90m e em todos os casos considerou-se 0% e 1% de ruídos de medição, β = 0,422, $h(x)/h_0$ = 0,75 e 25% de dano.

O caso de medições sem interferência de ruído, representa uma situação ideal. Utiliza-se esses resultados para mostrar a capacidade do método Alcateia em identificar danos, caso fosse usado algum tratamento para eliminar tais perturbações.

Com o objetivo de simular de forma mais realística os dados experimentais, a simulação da presença de ruído nas medições foi abordada, adicionando-se às formas modais de vibração uma perturbação aleatória com distribuição uniforme. Os modos de vibração, com imposição de ruído, são então fornecidos através da seguinte formulação,

$$\Phi_r(i,j) = \Phi(i,j)\left(1 + \frac{p}{100}\,\text{rand}(-1,1)\right), \tag{4.56}$$

onde Φ_r corresponde aos modos de vibração contaminados por ruído, Φ aos modos de vibração originais, p é o nível de ruído acrescentado e $\text{rand}(-1,1)$ é uma função geradora de números aleatórios no intervalo (−1,1). Devido a aleatoriedade dos métodos estocásticos, foram realizadas 10 simulações com o método Alcateia e o resultado é apresentado pela média aritmética dos resultados parciais.

Na Tabela 4.16 seguem os valores dos parâmetros que o método Alcateia utilizou para a resolução do problema sem a influência de ruídos. Com o critério de parada com tolerância de 10e(-4), as execuções foram interrompidas antes que todos os laços externos fossem executados.

Tabela 4.16: Valores dos parâmetros para danos sem ruídos.

Parâmetros	**Valor**
Número de variáveis	21
Número de Lobos	10
Número de Laços Externos	100

Número de Laços Internos	200
Coeficiente de Contração	0,01
Coeficiente de Independência	0,75
Tolerância	10e(-4)

Na Tabela 4.17 seguem os valores dos parâmetros utilizados para a resolução do problema com a influência de ruídos. Também foi utilizado critério de parada com tolerância de 10e(-4), logo as execuções foram interrompidas antes que todos os laços externos fossem executados.

Tabela 4.17: Valores de parâmetros para danos com ruídos.

Parâmetros	**Valor**
Número de variáveis	21
Número de Lobos	30
Número de Laços Externos	100*
Número de Laços Internos	700
Coeficiente de Contração	0,01
Coeficiente de Independência	0,8
Tolerância	10e(-4)

Nos resultados gráficos da identificação de danos, o eixo vertical representa a espessura relativa pela espessura nominal ($h(x)/h_0$) e o eixo horizontal o número da posição da viga.

4.4.2.1 Dano na posição 4

A Figura 4.23 mostra o dano na posição 4 e sem interferência de ruídos. Apesar da posição do dano ser próxima ao extremo da viga e por isto com menor sensibilidade à presença de dano, o método Alcateia foi capaz de localizar e quantificar o dano estrutural com precisão, com uma média de 31,2 iterações e com um tempo médio de execução de 46,2166s.

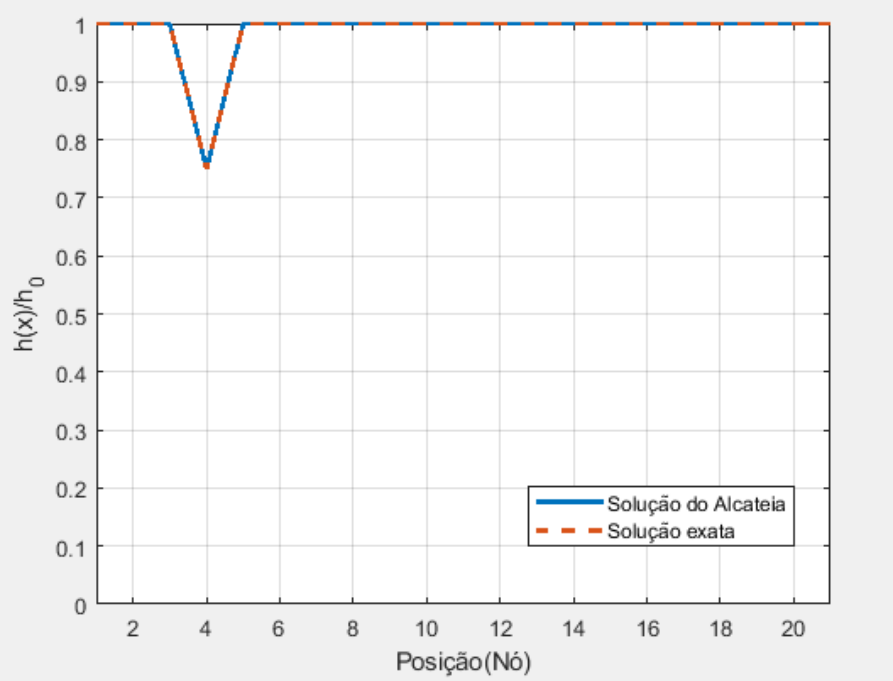

Figura 4.23: Dano na posição 4 sem ruído.

Com interferência de ruído, o método precisou de um tempo médio de 362,8541s e em média 32,5 iterações para chegar ao resultado apresentado na Figura 4.24. O método identificou corretamente a região do dano e quantificar em 15,10% de um dano imposto de 25% na quarta posição (Tabela 4.18).

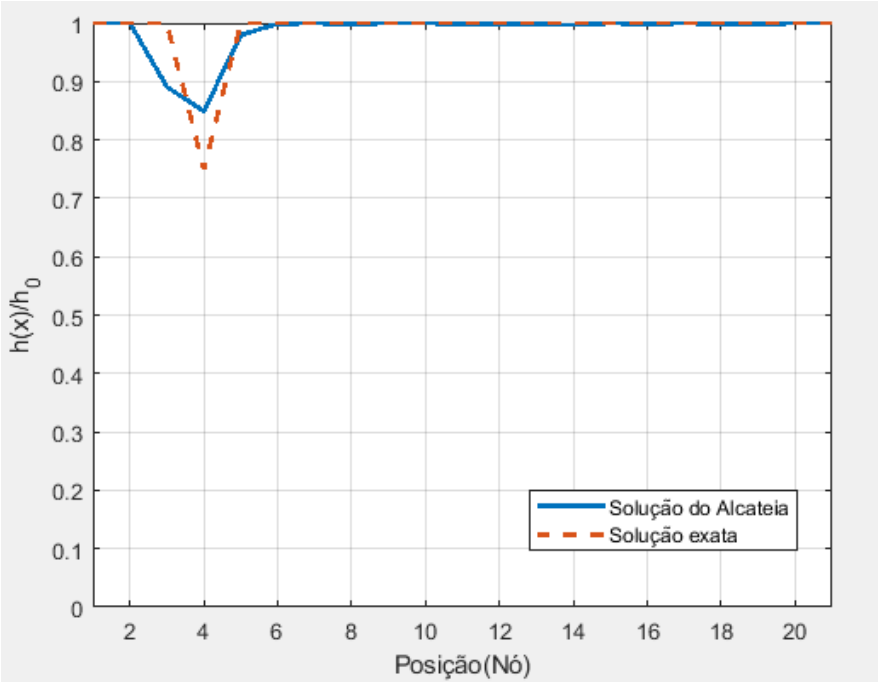

Figura 4.24: Dano na posição 4 com ruído.

Tabela 4.18: Dano na posição 4 e nas demais posições.

Posição	Dano
1	0,00%
2	0,00%
3	10,91%
4	15,10%
5	2,03%
6	0,09%
7	0,02%
8	0,16%
9	0,00%
10	0,00%
11	0,20%
12	0,18%
13	0,14%
14	0,33%
15	0,00%
16	0,17%
17	0,00%
18	0,15%
19	0,18%
20	0,00%
21	0,00%

4.4.2.2 Dano na posição 5

Considerando o dano na quinta posição e sem interferência de ruído, o método conseguiu localizar e quantificar o dano com exatidão (Figura 4.25). Foi necessário um tempo médio de 71,3675s e um média de 51,9 iterações.

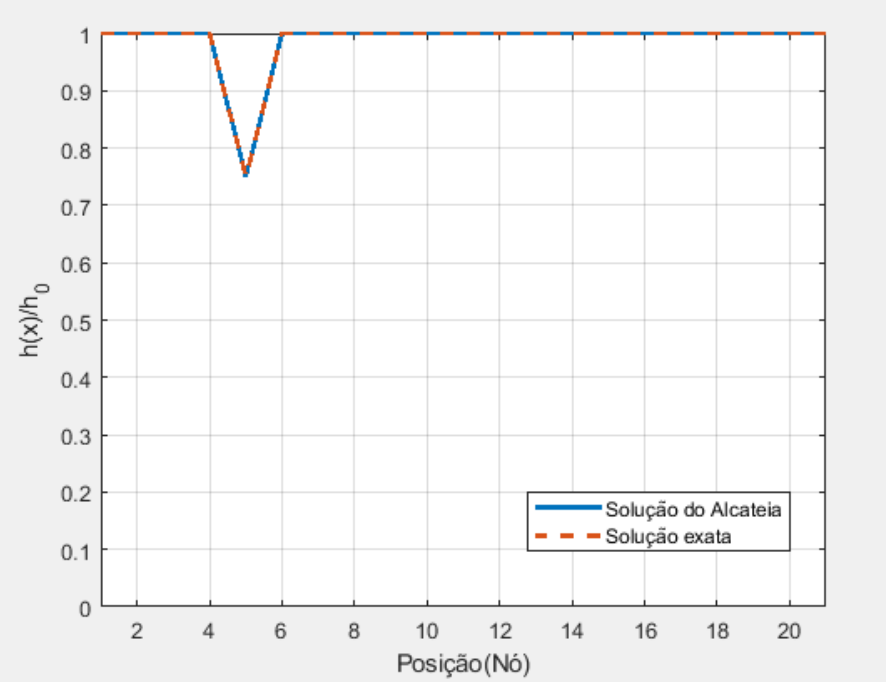

Figura 4.25: Dano na posição 5 sem ruído.

Na Figura 4.26 tem-se a resolução do método com interferência de ruído. Com um tempo médio de execução de 397,2705s e uma média de 23,9 iterações, o método conseguiu localizar a região do dano e quantificar em 14,76% de um dano de 25% na quinta posição (Tabela 4.19).

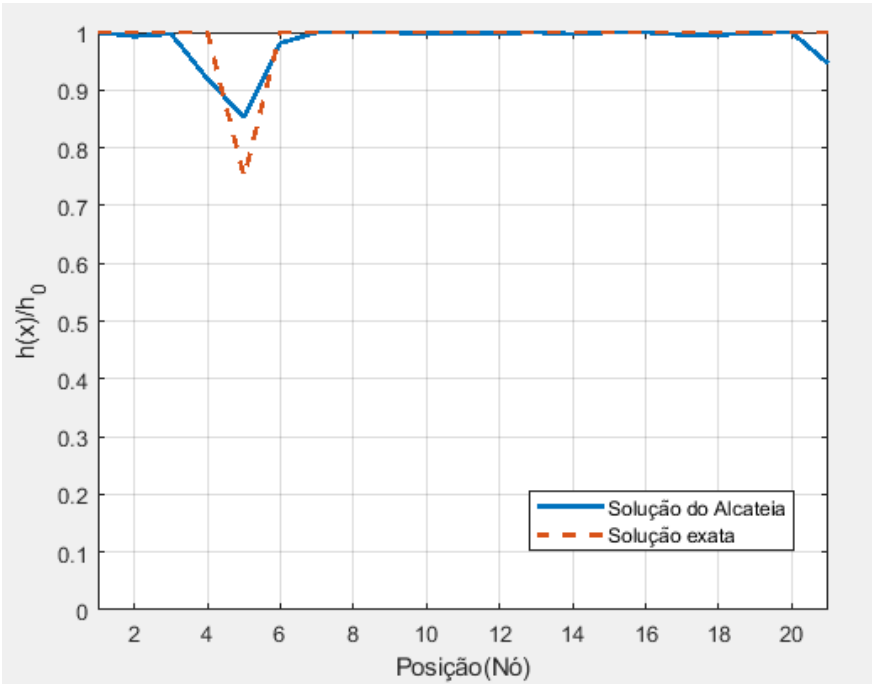

Figura 4.26: Dano na posição 5 com ruído.

Tabela 4.19: Dano na posição 5 e nas demais posições.

Posição	Dano
1	0,00%
2	0,68%
3	0,24%
4	8,06%
5	14,76%
6	1,86%
7	0,00%
8	0,00%
9	0,00%
10	0,19%
11	0,07%
12	0,13%
13	0,00%
14	0,29%
15	0,00%
16	0,00%
17	0,43%
18	0,44%
19	0,09%
20	0,00%
21	5,42%

4.4.2.3 Dano na Posição 8

Novamente pode-se notar na Figura 4.27 que o método Alcateia conseguiu localizar e quantificar, satisfatoriamente, o dano na posição 8, quando não foram considerados ruído de medição, com uma média de 35,9 iterações em um tempo médio de execução de 52,8414s.

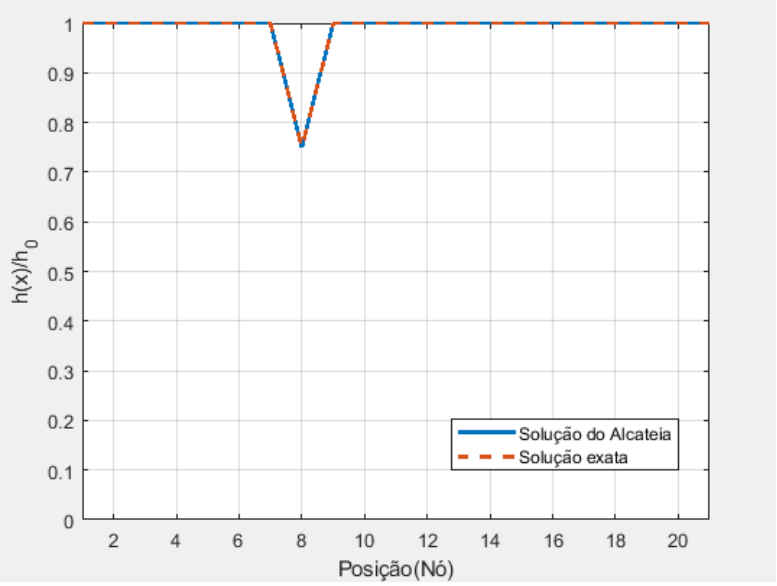

Figura 4.27: Dano na posição 8 sem ruído.

Apesar de interferência de ruído, o método Alcateia conseguiu localizar a região do dano e quantificar em 15,69% de um dano imposto de 25% (Figura 4.28), num tempo médio de execução de 395,8901s e uma média de 25,1 iterações. A Tabela 4.20 contém os resultados apresentado pelo método para o dano de 25% na posição 8, com interferência de ruído.

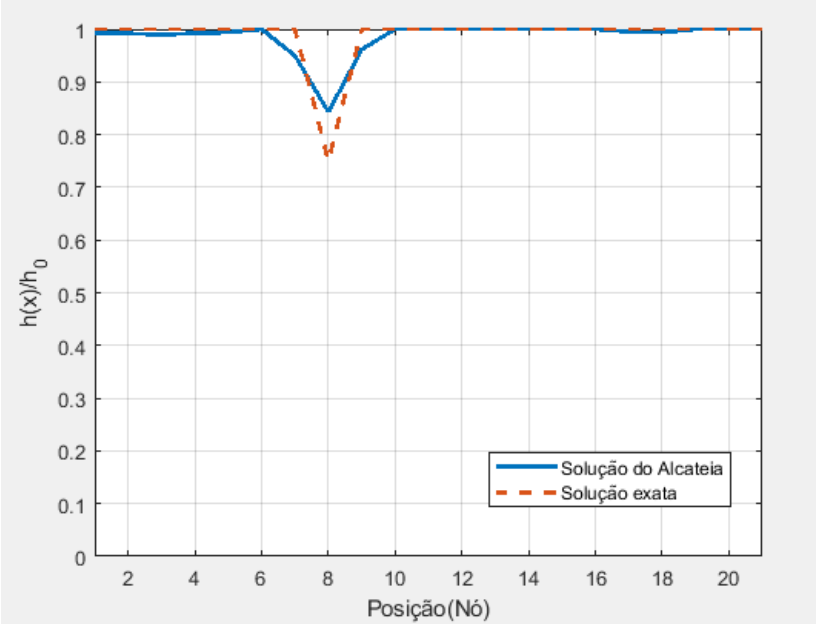

Figura 4.28: Dano na posição 8 com ruído.

Tabela 4.20: Dano na posição 8 e nas demais posições.

Posição	Dano
1	0,80%
2	0,78%
3	1,06%
4	0,74%
5	0,60%
6	0,00%
7	5,05%
8	15,69%
9	3,83%
10	0,00%
11	0,00%
12	0,00%
13	0,00%
14	0,00%
15	0,00%
16	0,00%
17	0,49%
18	0,61%
19	0,00%
20	0,00%
21	0,00%

4.4.2.4 Dano na posição 9

Considera-se, agora, um dano próximo da região central da viga, na nona posição, sem adição de ruídos. Pode-se verificar que o método Alcateia conseguiu localizar e quantificar o dano com bastante exatidão (Figura 4.29) em um tempo médio de execução de 85,8073s e com uma média de 69,9 iterações.

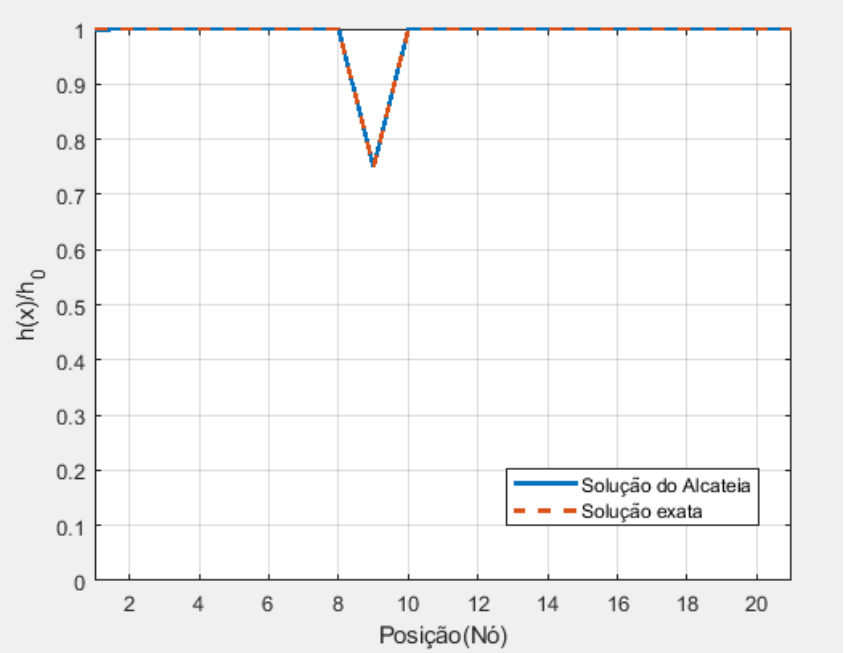

Figura 4.29: Dano na posição 9 sem ruído.

Com a adição de ruído, o método conseguiu localizar o dano na região correta e quantificar em 14,89% de um dano imposto de 25% (Figura 4.30), num tempo médio de execução de 428,4715s e uma média de 27,4 iterações. A Tabela 4.21 mostra os resultados apresentado pelo método para o dano de 25% na posição 9, com interferência de ruído.

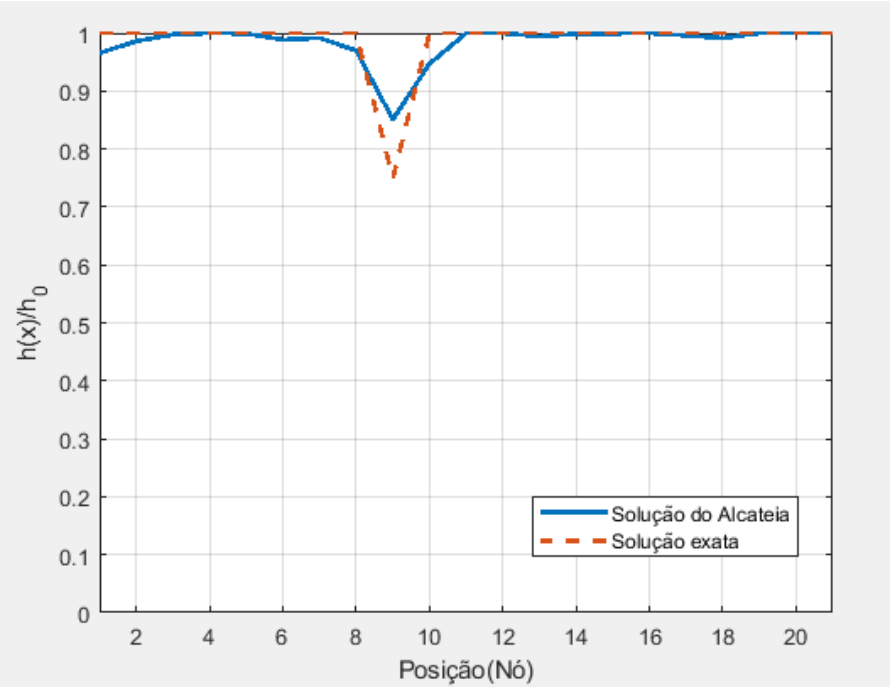

Figura 4.30: Dano na posição 9 com ruído.

Tabela 4.21: Dano na posição 9 e nas demais posições.

Posição	Dano
1	3,45%
2	1,38%
3	0,23%
4	0,00%
5	0,08%
6	1,15%
7	0,84%
8	3,01%
9	14,89%
10	5,29%
11	0,00%
12	0,00%
13	0,57%
14	0,13%
15	0,06%
16	0,00%
17	0,39%
18	0,88%
19	0,01%
20	0,00%
21	0,00%

4.4.2.5 Dano na posição 17

Na Figura 4.31 pode-se verificar que o método conseguiu localizar e quantificar satisfatoriamente o dano na décima sétima posição, quando não foram considerados ruído de medição, em um tempo médio de execução de 77,9941s e com uma média de 54,3 iterações.

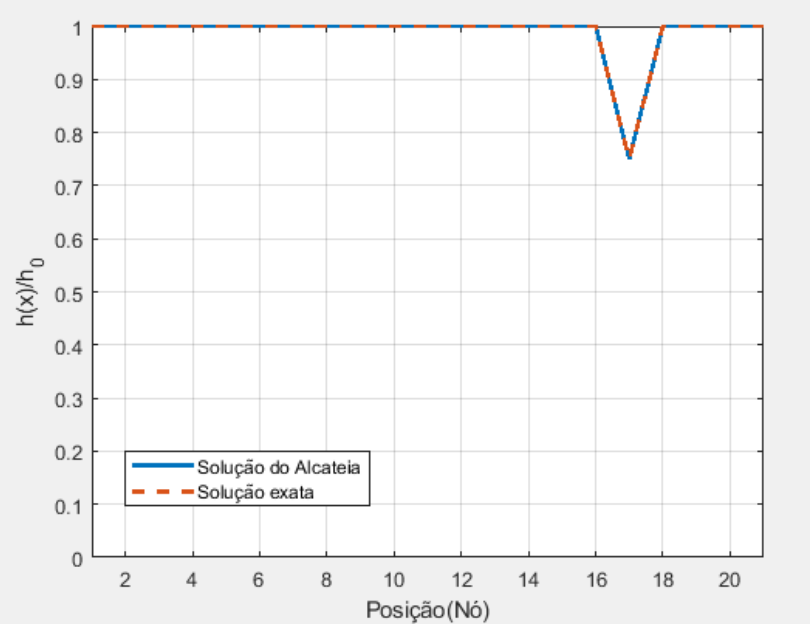

Figura 4.31: Dano na posição 17 sem ruído.

Tendo interferência de ruído, o método conseguiu localizar o dano na região corretamente e quantificar em 16,7% do 25% de dano (Figura 4.32) em um tempo médio de 427,9916s e com 22,1 iterações. A Tabela 4.22 contém os resultados apresentado pelo método para o dano de 25% na posição 17, com interferência de ruído.

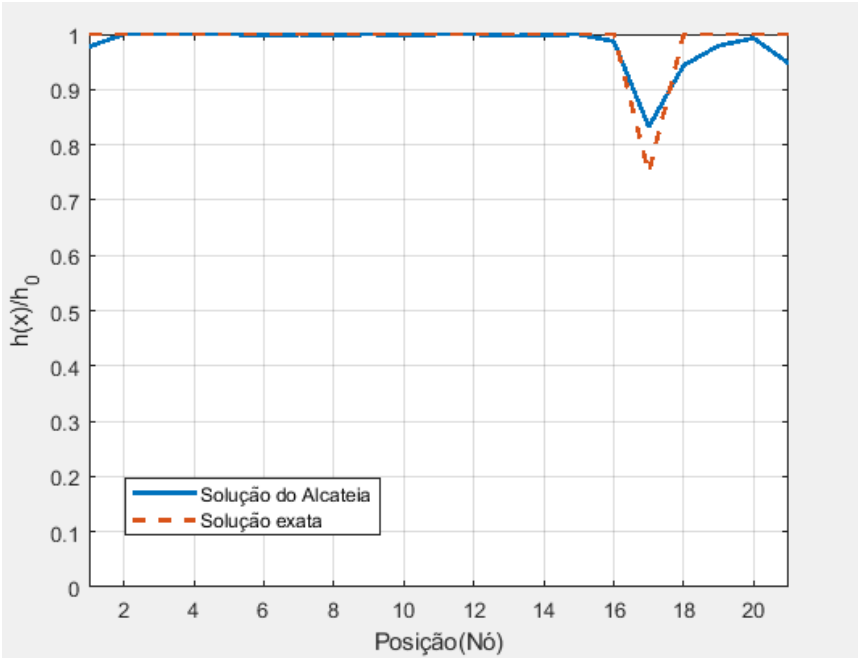

Figura 4.32: Dano na posição 17 com ruído.

Tabela 4.22: Dano na posição 17 e nas demais posições.

Posição	Dano
1	2,2%
2	0,0%
3	0,0%
4	0,0%
5	0,0%
6	0,2%
7	0,0%
8	0,2%
9	0,0%
10	0,2%
11	0,0%
12	0,0%
13	0,3%
14	0,2%
15	0,0%
16	1,3%
17	16,7%
18	5,6%
19	2,1%
20	0,7%
21	5,3%

4.4.2.6 Dano na posição 18

Na décima oitava posição, o método Alcateia localizou e quantificou corretamente o dano na estrutura (Figura 4.33), quando não foram considerados ruído de medição, em um tempo médio de execução de 45,3709s e com uma média de 29,4 iterações.

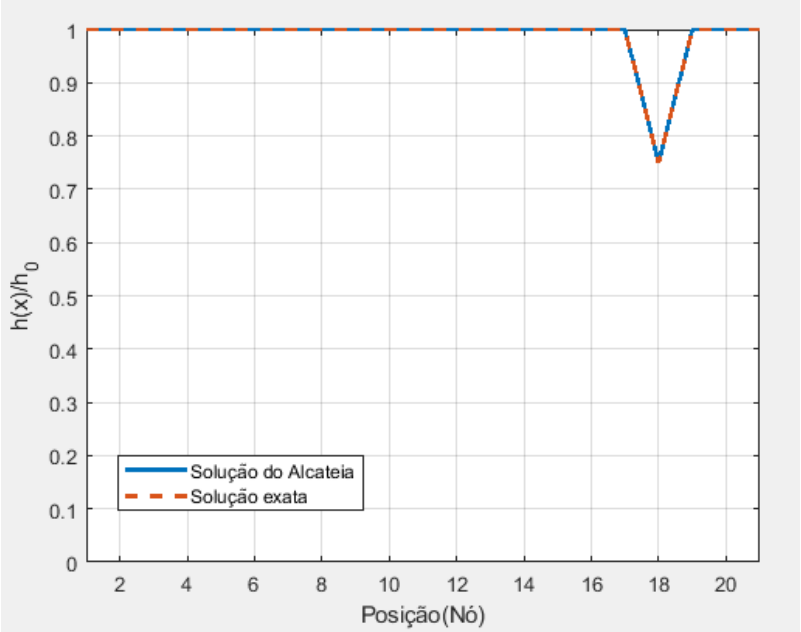

Figura 4.33: Dano na posição 18 sem ruído.

Com ruído, o método Alcateia conseguiu localizar a região do dano corretamente e quantificar em 16,3% do 25% de dano (Figura 4.34) em um tempo médio de 345,9402s e com 20,5 iterações. A Tabela 4.23 contém os resultados apresentado pelo método para o dano de 25% na posição 18, com interferência de ruído.

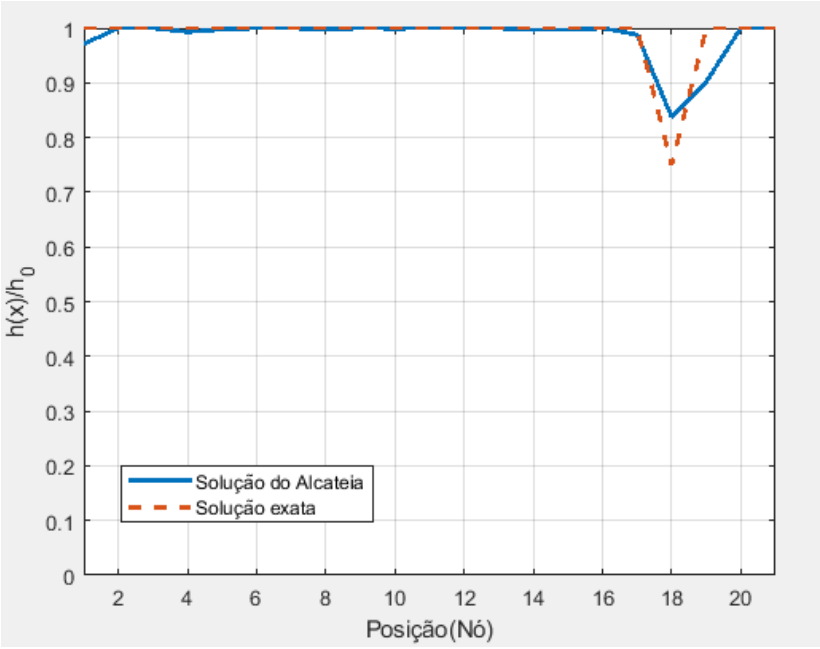

Figura 4.34: Dano na posição 18 com ruído.

Tabela 4.23: Dano na posição 18 e nas demais posições.

Posição	Dano
1	2,9%
2	0,0%
3	0,0%
4	0,7%
5	0,2%
6	0,0%
7	0,0%
8	0,2%
9	0,0%
10	0,1%
11	0,0%
12	0,0%
13	0,0%
14	0,3%
15	0,3%
16	0,0%
17	1,2%
18	16,3%
19	9,8%
20	0,0%
21	0,0%

Capítulo 5

5 Conclusões e trabalhos futuros

O presente estudo teve por objetivo geral compreender o conceito de métodos de otimização estocástico, buscando verificar e validar o método de otimização bio-inspirado Alcateia; e teve por objetivos específicos: testar o funcionamento do método para funções multimodal sem restrições; testar o funcionamento do método para funções multimodal com restrições; testar o funcionamento do método para projetos de engenharia e, por fim, testar o método para identificar dano em uma viga.

E ao fim do presente estudo pode-se dizer que os objetivos foram alcançados, pois os conceitos que se buscou compreender foram analisados e os testes foram realizados.

Foi possível perceber com o estudo que as técnicas de otimização são objetivos de estudo de muitos pesquisadores que a utilizam por serem técnicas que se mostram eficientes, por vezes, em alguns problemas e em outros não. Dessa forma, tem sido, ao longo dos tempos, desenvolvidos novos algoritmos de otimização que são processos de busca por uma solução que forneça máximos benefícios, ou seja, soluções ótimas. No entanto, há que se ressaltar que nem sempre o ótimo é alcançado, ainda, que esta seja a meta. Muitas vezes, o que se encontra é a melhor solução possível.

Verificou-se com o estudo que o problema de otimização pode ser interpretado como uma busca por valores variáveis que levem à maximização ou minimização de algumas funções dentro de determinados domínios. De modo geral, estes valores são definidos por meio de restrições tecnológicas, físicas ou normativas. Os processos de otimização, basicamente, são problemas matemáticos aplicados à parâmetros específicos.

Para o presente estudo, o que se propôs foi uma meta-heurística que tem como base o comportamento dos lobos. Há que se ressaltar que foram obtidos, com as funções testadas, ótimos resultados, o que deixou evidente seu potencial.

Optou-se pela utilização do método Alcateia por sua capacidade de resolução de problemas com inúmeros mínimos ou máximos locais, pois esta é uma

característica que pode ser encontrada em problemas encontrados em várias aplicações. o que termina por levar grande parte dos algoritmos de otimização a convergirem em direção de ótimos locais.

Nos testes realizados nas funções multimodais sem restrições e funções multimodais com restrições, o método encontrou, na maioria das funções, valores iguais aos valores analíticos, nas demais funções os valores encontrados foram muito próximos aos valores analíticos.

No que compete aos testes de engenharia, o que se verificou é que, neste cenário, o método Alcateia resolveu todos os problemas de forma satisfatória e encontrou novos valores de mínimo, quando comparado à literatura especializada, para quase todos os casos.

O método aplicado na identificação de danos de uma viga, conseguiu identificar a posição do dano e quantificar o valor deste dano, na ausência de ruídos, com precisão. Na presença de ruídos o método identificou a região potencialmente danificada e quantificou, numa proporção muito boa, o valor deste dano.

Sendo assim, buscou-se, com o presente estudo, ótimos em funções multimodais com restrição, funções multimodais sem restrição, projetos de engenharia e em identificação de danos em uma viga. E ficou comprovado, com os resultados apresentados, que o método Alcateia demonstrou potencial, inclusive, para problemas mais complexos.

Como sequência deste trabalho pretende-se avaliar o desempenho do algoritmo Alcateia em relação a busca de ótimos em problemas de contexto multi-objetivo, decrementar e incrementar quantidade de lobo, identificar danos em viga com imposição de dados ruidosos mais severos e com simulação de danos múltiplos.

Referências

ALFARES, F; ESAT, I. Real-coded quantum inspired evolution algorithm applied to engineering optimization problems. In: **Proceedings of the Second International Symposium on Leveraging Applications and For mal Methods, Verification and Validation**. Washimgton, USA. IEEE Computer Society, p. 169-176, 2006.

AL-MARZOUG, S. M.; HODGSON, R. J. Optics Communications. **Luus-Jaakola optimization procedure for multilayer optical coatings**, set., 2006.

ALVIN, K. F.; PETERSON, L. D; PARK, K. D. Method for determining minimum-order mass and stiffness matrices from modal test data. **AIAA Journal**, Vol. 1, n. 33, pp. 128–135, 1995.

BASTOS, E. A. **Otimização de Seções Retangulares de Concreto Armado Submetidas à Flexo-Compressão Oblíqua Utilizando Algoritmos Genéticos**. Rio de Janeiro, 2004.

BOITANI, L.; CIUCCI P. Comparative social ecology of feral dogs and wolves. **Ethology Ecology & Evolution**, volume 7, pages 49-72, 1995.

BRAJEVIC. I.; TUBA M.; SUBOTIC, M. Improved artificial bee colony algorithin for constrained problems. **Proceedings of the 11th WSEAS International Conference on Neural Networks, Fuzzy Systems and Evolutionary Computing**. Stevens Point, USA: WSEAS, p. 185-190, 2010.

BRANDÃO, M. A. **Estudo de Alguns Métodos Determinísticos de Otimização Irrestrita**. Uberlândia, 2010.

CAGNINA, L. C.; ESQUIVE, S. C. Solving engineering optimization problems with the simple constrained particle swarm Optimizer. **Advances in Engineering Software**, pp. 319-326, 2008.

CARRILLO, O. J. **Algoritmo híbrido para avaliação da integridade estrutural**: uma abordagem heurística.Tese de Doutorado, Universidade de São Paulo., 2007.

COELHO, C.A.C.; MONTES, E.M. Constraint-handling in genetic algorithms through the use Of dominance-based tournament selection. **Advanced Engineering Informatics**, 16, 193-203, 2002.

COELHO, A. C. R.; HENDERSON, H.; SACCO, W. F.; DOMINGOS, R. P. Comparasion of Luus-Jaakola algorithm and particle swarm optimization applied to a multimodal test function. **X Encontro de Modelagem Computacional**, Instituto Politécnico - UERJ, Nova Friburgo, Rio de Janeiro, 2017.

CORRÊA, C. A. J,; CORREA, R. A. P. . Alcateia - Um Novo Algoritmo de Otimização. In: **XXXVI Congresso Nacional de Matemática Aplicada e Computacional**, Gramado, 2016.

CORRÊA, R. A.; CORRÊA JR., C. A. Uma aplicação prática do método Alcateia em um problema de identicação de danos em uma viga. **Proceeding Series of the Brazilian Society of Computational and Applied Mathematics**, 2017.

CORRÊA, R.A.P. Identificação de danos em estruturas bi-dimensionais via matriz de flexibilidade baseada em um modelo de dano contínuo. 2013. 131 f. **Tese em Doutorado em Modelagem Computacional.** IPRJ/UERJ, Nova Friburgo, 2013.

DEB, K. Optimal design of a welded beam via genetic algorithms. **AIAA Journal**, 29 (ll), pp. 2013-2015, 1991.

DEB, K. GeneAS: a robust optimal design technique for mechanical component design. In: Dasgupta, D., Michalewicz„ Z. (Eds.), Evolutionary Algorithms in Engineering Applications. **Springer**, Berlin, pp. 497-514, 1997.

EGLESE, R.W. Simulated annealing: a tool for operation research. **European Journal of Operation Research**, v. 46, p. 271-281, 1990.

FERREIRA, F. D. S. **Uma abordagem numérico-experimental para a identicação de dano estrutural utilizando o método Simulated Annealing**. Dissertação de Mestrado, Escola de Engenharia, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2008.

GALEANO, J. G. D. **Aspectos numéricos do problema de cálculo de fenômenos de vaporização retrógrada dupla**. Dissertação de Mestrado, Instituto Politécnico, Universidade do Estado do Rio de Janeiro, Nova Friburgo, 2007.

GONÇALVES, A. R. **Otimização em ambientes dinâmicos com variáveis contínuas empregando algoritmos de estimação de distribuição**. Dissertação de Mestrado, Universidade Estadual de Campinas, Campinas, SP, 2011.

GUYAN, R. J. Reduction of stiffness and mass matrices. American Institute of Aeronautics and Astronautics **Journal**, vol. 3, n. 2, p. 380, 1965.

HE. Q.; WANG. L. An effective co-evolutionary particle swarm optimization for constrained engineering design problem. **Engineering Applications of Artificial Intelligence**, 20(1):89-99, 2007.

HOLTZ, G. C. C. **Traçado automático de envoltórias de esforços em estruturas planas utilizando um algoritmo evolucionário**. Dissertação de Mestrado, Pontifícia Universidade Católica do Rio de Janeiro. Rio de Janeiro, 2005.

IZMAILOV, A., SOLODOV, M., **Otimização – Volume 2: Métodos Computacionais.** Rio de Janeiro. IMPA, 2007.

JAMIL, M.; YANG, X. S. A literature survey of benchmark functions for global optimisation problems. **International Journal of Mathematical Modelling and Numerical Optimisation**, 4(2), 150, 2013.

JEZOWSKI, J.; BOCHENEK, R.; ZIOMEK, G. Random search optimization approach for highly multi-modal nonlinear problems. **Advances in Engineering Software**, 36, 504-517, 2015.

KANNAN, B.; KRAMER, S. An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. **Journal of Mechanical Design**, 1, 16(2):405-411, 1994.

KAVISKI, E., PRADO, A. L., CUMIN, L. M. G. Solução de sistemas de equações não-lineares pelo método do recozimento simulado. **XXXI Congresso Nacional de Matemática Aplicada e Computacional**, 370_375, 2008.

KENNEDY, J.; EBERHART, R. C. Proceedings of the International Conference on Neural Networks. **Piscataway**, NJ, V.4, pp. 1942 - 1948, 1995.

KIRKPATRIK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by Simulated Annealing. **Science**, 220, pp. 671–680, 1993.

KNUPP, D. C.; SILVA-NETO, A. J.; SACCO, W. F. Estimation of radiative properties with the Luus-Jaakola method. **X Encontro de Modelagem Computacional**. Instituto Politécnico - UERJ, Nova Friburgo, Rio de Janeiro, 2017.

LARA HERRERA, C. N. **Algoritmo de tomografia por impedância elétrica baseado em Simulated Annealing**. Dissertação de Mestrado, Escola Politécnica da Universidade de São Paulo. São Paulo, 2007.

LIMA JR, C. A. S., COSTA, J. S., SACCO, W. F. Comparação entre os algoritmos luus-jaakola e particle swarm optimization (pso) em dois problemas de otimização global. **XXXII Congresso Nacional de Matemática Aplicada e Computacional**, 873_879, 2009.

LOBATO, F. S.; SILVA-NETO, A. J.; STEFFEN JR., F. Um estudo comparativo usandoevolução diferencial e recozimento simulado aplicado a um problema inverso de transferência radiativa. **X Encontro de Modelagem Computacional**. Instituto Politécnico - UERJ, Nova Friburgo, Rio de Janeiro, 2007.

LOBATO, F. S.; STEFFEN JR., V. Solution of Optimal Control Problems using MultiParticle Collision. 9 th, Brazilian Conference on Dynamics. **Control and Their Applications**, DINCON-2010.

LOBATO, F. S.; ALMEIRDA, G. M.; FERNANDES, C. F.; ALMEIDA, L. M. S. Algoritmo de Colisão de Partículas Aplicado ao Projeto de Sistemas de Engenharia. **Nono Simpósio de Mecânica Computacional**, 26 a 28 de maio de 2010.

LUUS, R.; HENNESSY, D. Optimization of Fed-Batch Reactors by the Luus-Jaakola Optimization Procedure. **Ind. Eng. Chem. Res**., Vol. 38, pp.1948- 1955, 1999.

LUUS, R.; JAAKOLA, T.H.I. Optimization by direct search and systematic reduction of the size of search region. **American Institute of Chemical Engineers Journal (AIChE)**,19 (4): 760–766, 1973.

LUZ, E. F. P.; BECCENERI, J. C.; CAMPOS VELHO, H. F. A New Multi-Particle Collision Algorithm for Optimization in a High Performance Environment**. Journal of Computational Interdisciplinary Sciences**, 1, pp. 3–10, 2008.

MARTÍNEZ GONZÁLEZ, Y.; MARTÍNEZ RODRÍGUEZ, J.; SILVA NETO, A.; GOMES WATTS RODRIGUES, P. Estimation of open channels hydraulic parameters with the

stochastic particlecollision algorithm. **Journal of the Brazilian Society of Mechanical Sciences and Engineering**., pp. 69-77, 2014.

MARTINS, S. V. Melhor horário para o sistema CEFET Campos: Um aplicativo para automatizar a elaboração de horários das aulas. **Vértices**, 6:9-26, 2004.

MISHRA, S. **Some new test functions for global optimization and performance of repulsive particle swarm method***.* Munich Personal RePEc Archive, 2006.

OLIVEIRA, G. T.; SARAMAGO, S. F. Estratégias de evolução diferencial aplicadas a problemas de otimização restritos. **15º POSMEC** – Simpósio do Programa de Pós-Graduação em Engenharia Mecânica., 2005.

PAPADAKIS, S.; MARKAKI, M. An in depth economic restructuring framework by using particle swarm optimization. **Journal of Cleaner Production**., pp. 329-342, abr., 2019.

RAGSDELL, K. M.; PHILLIPS, D. T. Optimal design of a class of welded structures using geometric programming. **ASME Journal of Engineering for Industries**, 98 (3), 1021-1025, 1976.

RAO, S. S**. Engineering optimization: theory and practice**. 4 ed., New York, USA: Wiley, 2009.

ROSENBROCK, H. H. An automatic method for finding the greatest or least value of a function. **The Computer Journal**. 3 (3): 175–184, 1960.

SACCO, W.; OLIVEIRA, C. A New Stochastic Optimization Algorithm based on a Particle Collision Metaheuristic. **6th World Congresses of Structural and Multidisciplinary Optimization.**, 2005.

SARAMAGO, S.F.P. Métodos de Otimização Randômica: Algoritmos Genéticos e Simulated Annealing. **SBMAC**, São Carlos, v.6. p.3. 2003.

SARAMAGO, S. F.P.; PRADO, J. R. Otimização por Colônia de Partículas. In: **25º Congresso Nacional de Matemática Aplicada e Computacional**, São Paulo, 2005, v. 1, p. 1-6.

SERAPIÃO, A. B. Fundamentos de otimização por inteligência de enxames: uma visão geral. **Controle e Automação Sociedade Brasileira de Automática**., pp. 271-304, mar., 2009.

SILVA, C. K. F.; SOUZA, M. A. P.; SILVA, Z. E.; MARIANI, V. C.. Comparativo dos métodos de otimização na estimativa da difusividade efetiva de massa de cogumelos via problema inverso. **V Congresso Nacional de Engenharia Mecânica**, Salvador, 2008, p. 1-10.

SILVA, M. R. **Um novo método híbrido aplicado à solução de sistemas não lineares com raízes múltiplas**. Tese de Doutorado, Instituto Politécnico, Universidade do Estado do Rio de Janeiro, Nova Friburgo, 2009.

SILVA-NETO, A. J.; SOEIRO, F. J. C. P. Estimation of the phase function of anisotropic scattering with a combination of gradient based and stochastic global optimization methods, Proceedings. **5 th World Congress on Computational Mechanics**, Vienna, Austria, July, 7-12, 2012.

SMITH, D. W.; PETERSON R. O.; Houston D. B. Yellowstone after Wolves. **BioScience**, volume 53, pages 330-340, 2003.

SOLVE a constrained nonlinear problem, problem-based. **Mathworks**, 2020. Disponível em: <https://www.mathworks.com/help/optim/ug/solve-nonlinear-optimization-problem-based.html>. Acesso em: 06 set. 2019.

SURJANOVIC, S.; BINGHAM, D. **Virtual Library of Simulation Experiments**: Test Functions and Datasets, 2013.

TELLES, W. R.; RODRIGUES, P. P. G. W.; SILVA NETO, A. J. d. Automatic calibration of a simulator applied to a mountain river employing experimental data of rainfall and level - case study: D'antas stream, rj. **RBRH**, 21, 1, 143_151, 2016.

TOWNSEND, A. Constrained optimization in Chebfun. **Chebfun**, 2014. Disponível em:<http://www.chebfun.org/examples/opt/ConstrainedOptimization.html>. Acesso em: 03 set. 2019.

VIANA, P. E. S. **Métodos de Simulação Estocástica Aplicados em Problemas Não Lineares**. Dissertação de Mestrado, Universidade Federal Fluminense, Volta Redonda, 2017.

WEI, L.; ZHANG, Z.; ZHANG, D.; LEUNG, S. C. A simulated annealing algorithm for the capacitated vehicle routing problem with two-dimensional loading constraints. **European Journal of Operational Research**., pp. 843-859, mar. 2018.

ZHOU, Y; PEI, S. A Hybrid Co-evolutionary Particle Swarm Optimization Algorithm for Solving Constrained Engineering Design Problems. **Journal of Computers**, VOL. 5, NO. 6, JUNE 2010, 965-972, 2010.

ZÖRNIG, P. **Introdução à programação não linear**. Brasília: UNB, 2011.