UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS TRINDADE OU CENTRO TECNOLÓGICO
CURSO DE GRADUAÇÃO EM ENGENHARIA CIVIL

Pedro Antônio Roldão Simon

Utilização do Search Group Algorithm para a otimização dimensional, geométrica e topológica de treliças planas em Python

Florianópolis
2022

Pedro Antônio Roldão Simon

Utilização do Search Group Algorithm para a otimização dimensional, geométrica e topológica de treliças planas em Python

Trabalho de Conclusão de Curso do Curso de Graduação em Engenharia Civil do Campus Trindade ou Centro Tecnológico da Universidade Federal de Santa Catarina para a obtenção do título de bacharel em Engenharia Civil.
Orientador: Prof. Rafael Holdorf Lopez, Dr.

Florianópolis
2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Simon, Pedro Antônio Roldão
Utilização do Search Group Algorithm para a otimização dimensional, geométrica e topológica de treliças planas em Python / Pedro Antônio Roldão Simon ; orientador, Rafael Holdorf Lopez, 2022.
92 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Engenharia Civil, Florianópolis, 2022.

Inclui referências.

1. Engenharia Civil. 2. Algoritmo de otimização. 3. Otimização estrutural. 4. Treliça Plana. 5. SGA. I. Lopez, Rafael Holdorf. II. Universidade Federal de Santa Catarina. Graduação em Engenharia Civil. III. Título.

Pedro Antônio Roldão Simon

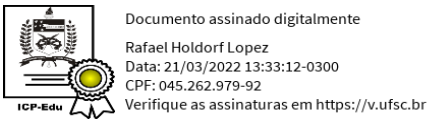
Utilização do Search Group Algorithm para a otimização dimensional, geométrica e topológica de treliças planas em Python

Este Trabalho Conclusão de Curso foi julgado adequado para obtenção do Título de "bacharel em Engenharia Civil" e aprovado em sua forma final pelo Curso de Graduação em Engenharia Civil

Florianópolis, 10 de março de 2022.

Profa. Liane Ramos da Silva, Dra.
Coordenadora do Curso

Banca Examinadora:



Prof. Rafael Holdorf Lopez, Dr.
Orientador
Universidade Federal de Santa Catarina

Prof. Leandro Fleck Fadel Miguel, Dr.
Avaliador
Universidade Federal de Santa Catarina

Thiago Moreno Fernandes, Me.
Avaliador
Universidade Federal de Santa Catarina

Este trabalho é dedicado aos meus queridos pais e à minha família.

AGRADECIMENTOS

Agradeço, primeiramente à minha família por todo o apoio que sempre me deram nas minhas decisões mais importantes e nos momentos mais difíceis. O resultado deste trabalho e a finalização de mais esse ciclo só foi possível por causa de vocês. Agradeço à minha mãe, Cristine, meu pai, José Antônio, e meu irmão, Tiago. Um agradecimento especial à minha irmã, Ana Paula, que se empenhou na revisão de todo o texto deste trabalho. Não teria conseguido sem sua grande ajuda.

A meu professor orientador, Rafael H. Lopez, por todas as conversas e materiais disponibilizados ao longo deste trabalho, e por ter possibilitado o meu aprendizado em um tema pouco visto ao longo da graduação, os algoritmos metaheurísticos. Um conhecimento que sem dúvidas me abre novas possibilidades na engenharia.

A todos os professores e servidores da UFSC, que direta ou indiretamente tiveram impacto em minha formação. Não tenho palavras para descrever meu desenvolvimento pessoal e profissional ao longo dos anos que frequentei essa universidade.

Por fim, sou imensamente grato aos colegas de classe que logo se tornaram grandes amigos. Saibam que vocês foram inspirações para mim. Obrigado Julia, Carol, Chico, Chon, Daniel, André, Marcos, João, Gabriel, Igor, Lucas e Oliver. Contem sempre comigo.

"If a machine is expected to be infallible,
it cannot also be intelligent."
(Alan Turing, 1947)

RESUMO

Neste trabalho, um algoritmo metaheurístico SGA (Search Group Algorithm) foi desenvolvido na linguagem computacional *Python* para a redução do peso de treliças planas. Foi aplicada a otimização simultânea de dimensão, geometria e topologia das estruturas. O algoritmo de análise estrutural foi desenvolvido com base no método da rigidez direta considerando as seguintes restrições: tensão admissível, flambagem, deslocamento dos nós da estrutura e validade da estrutura. O algoritmo foi aplicado para resolução de cinco problemas de otimização de treliças planas já estudados na literatura. Um espaço amostral com 100 resultados foi coletado para cada problema com intuito de medir a eficiência do algoritmo desenvolvido em relação a outros autores. Os resultados deste estudo superaram outras literaturas para dois problemas aplicados, uma treliça de 11 barras e uma treliça de 39 barras. Para outros dois problemas foram encontrados resultados equivalentes aos outros autores. Devido aos resultados satisfatórios deste estudo, o algoritmo aplicado pode ser classificado como competitivo.

Palavras-chave: Algoritmo de otimização. Metaheurístico. Otimização estrutural. Treliça plana. SGA.

ABSTRACT

On this study, a SGA (Search Group Algorithm) meta-heuristic algorithm was developed using Python programming language for weight reduction of planar trusses. A simultaneuous optimization of size, shape and topology was applied. The structural analysis algorithm was developed based on the direct stiffness method, taking into account the following constraints: admissible stress, buckling, node displacement and validity of the structure. The algorithm was applied on the resolution of five structural problems involving the optimization of planar trusses previously studied in the literature. A sample space of 100 results was gathered for each problem in an effort to measure the efficiency of the developed algorithm compared to results obtained by different authors. The results of the current study surpass those obtained by others on two of the problems tested, namely an 11-bar truss and a 39-bar truss. For two other problems, the results found were equivalent to those obtained by other similar studies. In view of the satisfactory results achieved by this study, the algorithm applied may be considered up to par.

Keywords: Optimization algorithm. Metaheuristics. Structural optimization. Planer truss. SGA.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

No atual cenário econômico, social e ambiental, a sustentabilidade coloca-se como uma questão cada vez mais urgente. Não é diferente na engenharia civil, onde o uso racional de recursos vem se tornando um novo pilar a ser considerado pelos profissionais da área. Por outro lado, a economia de recursos é também uma forma de tornar as empresas mais competitivas no mercado, otimizando o gasto de materiais e mão de obra e oferecendo preços mais competitivos. Dessa forma, é claro o apelo para a utilização de métodos avançados de otimização no campo da engenharia.

A aplicação de algoritmos de otimização em problemas de engenharia estrutural é relativamente recente. Segundo DEDE et al. (2019), isso se deve à grande complexidade dos modelos matemáticos, o que acaba por gerar um espaço de soluções não-lineares e não-convexas, tornando difícil e custosa sua implementação. Porém, com a proposição de novos métodos de otimização avançados e o aumento do poder computacional das últimas décadas, essas técnicas estão sendo cada vez mais utilizadas em projetos de larga escala na engenharia civil. Esses modelos de otimização vêm sendo pesquisados e utilizados em diversos campos da engenharia, como transporte, planejamento e gerenciamento de obras, engenharia geotécnica e engenharia estrutural.

É nesse cenário que entram os algoritmos metaheurísticos, que propõem uma forma iterativa de lidar com problemas não-lineares e não-convexos da engenharia. Os algoritmos metaheurísticos utilizam ideias estocásticas ou probabilísticas, que são intuitivas ou baseadas em tentativa e erro (iterativas) e que tendem a encontrar a solução ótima, mas nem sempre (RAO, 1996). Assim, essas técnicas são novas abordagens em relação à área de programação matemática tradicional, que geralmente se utilizava de gradientes da função objetivo para encontrar a solução ótima.

Entre os primeiros algoritmos de otimização metaheurísticos desenvolvidos e que tiveram sucesso na aplicação em diferentes problemas robustos, é possível citar os Algoritmos genéticos (GA) por Holland (1975), Recuo Simulado (SA) por Kirkpatrick, C. D. Gelatt e Vecchi (1983), Tabu Search (TS) por Glover (1977), Otimização por enxame de partículas (PSO) por Kennedy e Eberhart (1995), Busca Harmônica (HS) por Geem e Kim (2001) e Teaching–Learning-Based Optimization (TLBO) por Rao, Savsani e Vakharia (2011). Esses algoritmos construíram a base para as áreas de aprendizado de máquina e otimização.

A aplicação dessas técnicas de otimização avançadas encontrou um campo fértil na engenharia de estruturas. Problemas de otimização estruturais normalmente possuem um vasto espaço de soluções, contendo variáveis que podem envolver a escolha de materiais, dimensão, topologia e geometria da estrutura, não sendo viável testar todas essas possibilidades. Em tal cenário, o objetivo passa a ser a produção de boas soluções em uma escala de tempo aceitável. Por sua vez, os metaheurísticos podem ser uma forma

eficiente de utilizar tentativa e erro para garantir soluções em um tempo prático e razoável (GANDOMI et al, 2013).

Segundo CHRISTENSEN (2008), os problemas de otimização estrutural podem ser divididos em três classes: **dimensional**, **geométrica** e **topológica**. A otimização **dimensional**, para o caso de uma treliça, resulta da variação das seções transversais das barras. A otimização **geométrica ou de forma** resulta da variação das posições dos nós estruturais, porém não muda a conectividade da estrutura. Já a otimização **de topologia ou de layout** resulta da exclusão ou inclusão de barras na estrutura. A utilização consecutiva dessas três classes de otimização aumenta a complexidade do problema, porém permite que o algoritmo explore um espaço maior de solução, entregando resultados difíceis de serem previstos manualmente. Vale destacar também que, de maneira geral, as funções objetivo a serem minimizadas em um problema de otimização estrutural são peso, rigidez, carga crítica e custo de produção.

Este trabalho utilizará o Search Group Algorithm (SGA), um algoritmo metaheurístico desenvolvido por Gonçalves, Lopez e Miguel (2015), para realizar a otimização de problemas de treliças planas, objetivando a redução de peso das mesmas. Os três tipos de otimização (topológica, geométrica e dimensional) serão aplicados simultaneamente, de forma a testar a eficácia do SGA e do algoritmo desenvolvido neste estudo para resolver problemas de treliças com elevado espaço de busca. Devido à natureza estocástica dos algoritmos metaheurísticos, um campo amostral com o resultado de 100 execuções do algoritmo será gerado para realizar comparações de validade estatística com outros autores que utilizaram diferentes algoritmos metaheurísticos para os mesmos problemas.

O algoritmo SGA e as rotinas de análise estrutural foram desenvolvidos na linguagem computacional *Python*. O algoritmo metaheurístico foi adaptado neste estudo para suprir as necessidades deste estudo. As rotinas de análise estrutural foram desenvolvidas através do método de matriz de rigidez direta para uma análise estrutural de primeira ordem elástica.

1.1 OBJETIVOS

Nas seções abaixo estão descritos o objetivo geral e os objetivos específicos deste TCC.

1.1.1 Objetivo Geral

Aplicação do algoritmo de otimização SGA para a redução do peso de treliças planas utilizando variáveis dimensionais, geométricas e topológicas e comparação dos resultados obtidos com outros estudos na literatura.

1.1.2 Objetivos Específicos

- Apresentar o cenário geral do uso de algoritmos metaheurísticos na engenharia estrutural
- Apresentar o funcionamento do algoritmo de otimização utilizado
- Desenvolver uma rotina de análise estrutural em *Python* por meio do método da matriz de rigidez direta
- Realizar a otimização dos problemas escolhidos, coletando os resultados obtidos
- Fazer uma análise de desempenho do algoritmo por meio de gráficos e tabelas e através de comparação com outros algoritmos de otimização.

1.2 JUSTIFICATIVA QUANTO À RELEVÂNCIA DO TEMA ABORDADO

A utilização de métodos de otimização avançados na engenharia estrutural é de extrema relevância na cadeia da construção civil. Em primeiro lugar, em um cenário global onde a sustentabilidade deve urgentemente ser atrelada à engenharia, é essencial a economia de recursos, sobretudo na engenharia civil, onde há grande desperdício de materiais. Em segundo lugar, o processo de produção de projetos na construção civil está passando por uma grande transformação com as metodologias BIM, tornando diversas etapas do projeto cada vez mais digitais. Assim, é conveniente o desenvolvimento de soluções computacionais que ampliem a capacidade do engenheiro de encontrar soluções inéditas de forma rápida. Por fim, a escolha da utilização da linguagem *open source* (de fonte aberta) *Python*, facilita o acesso ao código por interessados que queiram explorar a área e as soluções desenvolvidas neste estudo.

2 OTIMIZAÇÃO

Ao longo da história, a humanidade buscou formas de atingir seus objetivos de maneira cada vez menos dispendiosa, seja reduzindo o tempo empregado, o uso de recursos ou o esforço físico e mental envolvidos. Com as revoluções científica e industrial, esse processo tornou-se cada vez mais relevante, e a ele é dado o nome de *otimização*.

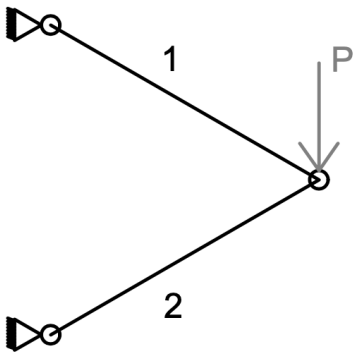
Problemas de otimização têm sido estudados há muito tempo por matemáticos e outros cientistas. Durante as últimas quatro décadas, com o desenvolvimento computacional e a criação de novos métodos de otimização, campos como a medicina, a biologia, a administração e a engenharia têm experimentado bons resultados na aplicação e resolução desse tipo de problema. Encontrar o trajeto mais curto de um ponto a outro, diminuir o custo operacional de uma infraestrutura ou reduzir o peso de uma estrutura são todos problemas de otimização, que podem ser resolvidos de forma esquemática por uma variedade de métodos.

Para RAO (1996, p.1, tradução nossa):

> Otimização é o ato de obter a melhor solução de acordo com condições pré-determinadas. No projeto, construção ou manutenção de qualquer sistema de engenharia, os engenheiros devem tomar decisões em múltiplos estágios. O objetivo dessas decisões é sempre o de minimizar os esforços necessários ou de maximizar os benefícios desejados. Assim, desde que esse objetivo possa ser definido através de uma função, a otimização é o processo de encontrar as condições que retornem o valor máximo ou mínimo desta função.

Já para Geem (2009), o problema de otimização é o processo de encontrar a melhor solução viável em meio a um conjunto que contenha todas as soluções. Esse conjunto descrito por Geem (2009) é o domínio do problema de otimização. De forma geral, qualquer problema de otimização pode ser discretizado por meio de alguns componentes. Pega-se como exemplo o caso de otimização de uma treliça plana simples com apenas duas barras, definida abaixo:

Figura 1 – Exemplo de treliça de duas barras a ser otimizada



Fonte: Autor

Com base neste problema, são definidos alguns componentes principais:

- **Função Objetivo**: a função que entrega um valor relacionado ao objetivo da otimização para ser minimizado ou maximizado. Valores comuns são distância, tempo, custo monetário, quantidade de material, peso, entre outros. No caso da treliça exemplificada acima, pode-se otimizar o seu peso através da função objetivo $f(x) = (A_1 \,.\, L_1 + A_2 \,.\, L_2) \,.\, \rho$. Onde ρ é a massa específica do material da estrutura e A_i e L_i são respectivamente a área da seção transversal e o comprimento da barra i.

- **Variáveis de Projeto**: são os parâmetros que descrevem as diferentes configurações possíveis do problema e são inseridos na função objetivo. Para o exemplo descrito, pode-se realizar a otimização do peso da estrutura a partir da mudança nas áreas da seção transversal da estrutura. Assim, teremos um conjunto de variáveis $\mathbf{x} = [x_1, x_2]$. Onde x_1 e x_2 são as áreas da seção transversal dos elementos 1 e 2 da treliça.

- **Restrições**: são regras impostas às variáveis de projeto para limitar soluções inviáveis ou que não sejam permitidas no modelo. No exemplo proposto é interessante que após a escolha das seções transversais das barras da estrutura, cada barra seja avaliada conforme um valor de tensão admissível σ_{adm}, impedindo que estruturas inviáveis sejam escolhidas no processo.

- **Domínio**: conjunto de todas as possíveis combinações de valores das variáveis de projeto, isto é, o espaço de busca. No exemplo, o domínio pode ser reduzido ao atribuir restrições de limite superior e inferior para as variáveis de seção transversal da treliça, delimitando uma faixa de possíveis valores a serem buscados no processo de otimização.

Sendo assim, um problema de otimização deve encontrar um conjunto de variáveis de projeto para minimizar sua função objetivo, sujeito a limitações impostas pelas restrições. A definição matemática de um problema de otimização segue o padrão:

Encontrar,

$$\mathbf{x} = [x_1, x_2, ..., x_n]$$

Para minimizar,

$$f_{obj} = f(\mathbf{x}) \tag{1}$$

Sujeito a:

$$g_i(\mathbf{x}) \leq 0;$$
$$h_j(\mathbf{x}) = 0;$$

Por meio dessa discretização simples, é possível descrever qualquer problema de otimização. É válido ressaltar que as restrições podem ser divididas em dois grupos. As **restrições de desigualdade** são sempre representadas por $g(\mathbf{x})$ e impõem uma regra de desigualdade a determinada propriedade do problema. Por padrão, as equações são sempre transformadas para a conotação $g(\mathbf{x}) \leq 0$. Já as **restrições de igualdade** são sempre representadas por $h(\mathbf{x})$, impondo a igualdade de uma propriedade do problema.

Ademais, segundo Goldberg (1989), otimização é a busca de um incremento de performance em direção a um ponto ótimo. Nesta definição, existem dois conceitos importantes: a busca por melhorias para (i) se aproximar de um (ii) ponto ótimo. Normalmente, os métodos de otimização tradicionais têm seu foco na convergência, não dando a devida atenção à própria eficiência do método, ou seja, define-se o ponto ótimo e avalia-se se o método atinge ou não este valor. Entretanto, quando se analisa a tomada de decisão de um humano, a qualidade da sua decisão é julgada com base no tempo e nos recursos disponíveis naquele cenário, além de compará-la a outras decisões já tomadas.

Quando um executivo toma uma decisão, por exemplo, deve-se julgá-la com base nas empresas concorrentes. Ele desenvolveu uma ferramenta melhor que seus competidores? Conseguiu atacar o mercado de forma mais eficiente? Quanto tempo levou para sobrepor as vendas dos competidores? Quando se exploram problemas de otimização no mundo real (maior complexidade), é comum que o ponto ótimo seja desconhecido e que os recursos sejam limitados. Assim, é importante que a qualidade do método de otimização seja sempre relacionada ao quão rápido este consegue obter soluções factíveis e satisfatórias com relação a outros.

Através dessa breve análise da definição de Goldberg (1989), já se vislumbra a insuficiência dos algoritmos tradicionais para diversas aplicações no mundo real. Por esse motivo, criou-se uma demanda por novos métodos mais robustos e versáteis, os quais serão discutidos a seguir.

2.1 ALGORITMOS METAHEURÍSTICOS

Ao longo do tempo, diversos métodos de otimização foram postulados e aplicados para diferentes problemas. Segundo Goldberg (1989), os métodos de otimização podem ser divididos em três grupos:

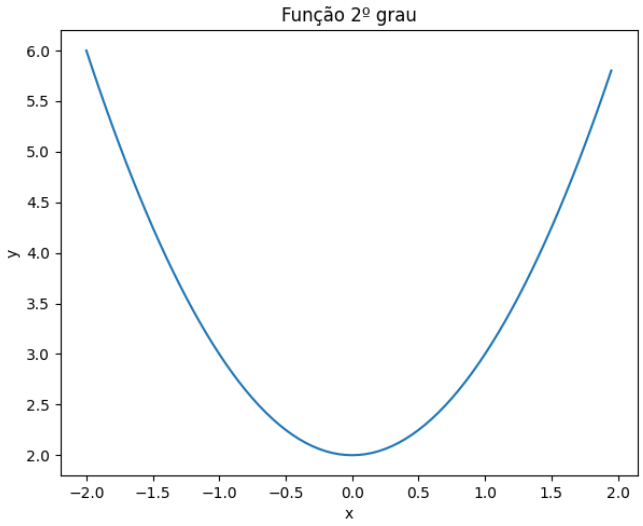
1. **Baseados em Gradientes**
2. **Enumerativos**
3. **Metaheurísticos**

Os métodos de otimização **baseados em gradientes** são aqueles que utilizam derivadas e gradientes da função para encontrar seus pontos de mínimo e máximo. Estes podem ser divididos em diretos (definem o gradiente da função como zero, encontrando

pontos de extremo) e indiretos (movimentam-se dentro da função na direção do seu gradiente local, até encontrar um ponto de extremo).

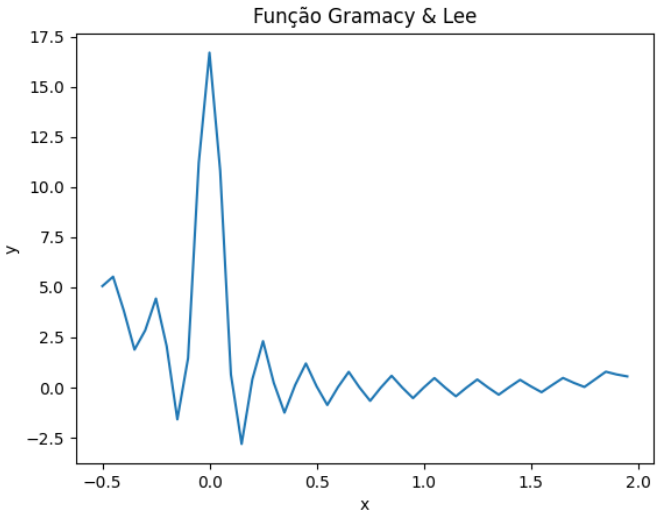
Contudo, o método de escalada por gradiente encontra diversos problemas ao ser aplicado em funções com muitos mínimos e máximos, sendo geralmente eficiente apenas na busca de ótimos locais. Na figura 2, por exemplo, essa solução não teria problemas para encontrar o ponto mínimo. Já na figura 3, cuja função possui muitos valores de mínimo, o resultado da otimização depende do ponto inicial escolhido, sendo provavelmente um mínimo local.

Figura 2 – Exemplo de função com apenas um mínimo global



Fonte: Autor

Figura 3 – Exemplo de função com muitos mínimos locais



Fonte: Autor

Além disso, em grande parte dos problemas reais, as funções objetivo não possuem continuidade em todo o seu domínio (não possuindo gradiente) ou são *não-convexas*. Uma função convexa pode ser definida como uma função cujo segmento de reta AB entre quaisquer dois pontos do domínio está sempre acima da própria função. É por isso que, na maioria dos casos, os métodos baseados em gradientes tornam-se inviáveis de serem aplicados em problemas que não atendem às necessidades matemáticas citadas.

Os **métodos enumerativos** são bastante simples e diretos. A ideia é que, dentro de um espaço de busca finito, o algoritmo procure na função objetivo os valores de todos os pontos do espaço, um de cada vez. Esse método aproxima-se bastante do estilo humano de busca, porém existem grandes limitações com relação ao tamanho do domínio da função. Normalmente, as funções objetivo tendem a possuir um espaço de busca muito grande para serem testadas em cada ponto do seu domínio, sendo ineficiente e impraticável a utilização de métodos enumerativos.

Por fim, os **algoritmos metaheurísticos** são mais recentes e são, em geral, um método iterativo que se utiliza da geração de novas buscas de forma aleatória. Apesar desta característica dos algoritmos metaheurísticos, é errado pensar que sua busca é completamente aleatória. Na verdade, a escolha de novos pontos aleatórios é realizada de forma direcionada, de maneira a tender a regiões do domínio que possuem uma boa chance de gerar melhores resultados. O uso da aleatoriedade contida nos algoritmos metaheurísticos pode parecer estranha inicialmente, porém, assim como na natureza, ela auxilia na eficiência do processo de otimização.

O termo "metaheurístico"foi cunhado por Glover (1986) e tem sua origem em duas palavras gregas. *Heuriskein*, que significa "a arte de encontrar novas estratégias para

resolver problemas", e *meta*, que significa "um maior nível de abstração". Antes desse termo ser cunhado, esse tipo de método era chamado de "heurístico moderno".

Para Kaveh (2017, p.2, tradução nossa):

> Um metaheurístico é formalmente definido como um processo de geração iterativo que orienta um heurístico subordinado a combinar de forma inteligente diferentes conceitos de **busca global** [*exploration*] e **busca local** [*exploitation*] para encontrar com eficiência soluções quase ótimas. Algumas estratégias de aprendizado podem ser aplicadas para conferir "inteligência"àqueles heurísticos de busca subordinados.

Na definição acima, dois importantes conceitos da otimização são estabelecidos. No inglês, o termo ***exploration*** refere-se à busca global, isto é, a exploração de todo o domínio da função, buscando diferentes regiões com resultados satisfatórios. Essa característica permite que o método de otimização não fique restrito a mínimos locais, conseguindo bom desempenho em problemas robustos. Já o termo ***exploitation*** refere-se à busca local, isto é, a exploração apenas das melhores regiões encontradas, permitindo refinar a solução final ao buscar os mínimos de cada região. O sucesso na aplicação desses dois conceitos é um dos pontos que explicam a eficiência dos métodos metaheurísticos na resolução de problemas robustos.

Os metaheurísticos não garantem que soluções quase ótimas serão encontradas de forma rápida para todos os problemas de otimização. Entretanto, esses algoritmos complexos encontram soluções quase ótimas para uma grande variedade de aplicações. Esta é a característica mais atrativa dos metaheurísticos, segundo Gonzalez (2007). É por isso que os metaheurísticos passaram a ser intensamente estudados e aplicados nas últimas duas décadas em diferentes áreas, tais como engenharia, física, química, artes, economia, marketing, genética, robótica, ciências sociais e políticas.

O funcionamento de grande parte dos metaheurísticos é inspirado em fenômenos e processos que ocorrem na natureza. Os primeiros e mais famosos algoritmos metaheurísticos desenvolvidos são o *Tabu Search - TS*, *Simulated Annealing - SA* (Recuo Simulado) e *Algoritmos Genéticos - GA*.

O Tabu Search é um método iterativo utilizado para resolver problemas combinatórios discretos. Glover (1977) foi quem originalmente sugeriu o método. O algoritmo é baseado na exploração de um espaço de busca que contém todas as soluções factíveis utilizando uma sequência de movimentos. A mudança de uma solução para outra resulta na melhor solução disponível. A palavra *Tabu* foi escolhida pois o algoritmo classifica alguns movimentos como proibidos (ou Tabu), com o objetivo de escapar dos ótimos locais e evitar prender-se em um ponto do espaço de busca. Os movimentos Tabu são formulados com base no histórico de movimentos, isto é, durante o processo de otimização, o algoritmo aprende a evitar alguns movimentos já utilizados.

O Simulated Annealing (Recuo Simulado) é um algoritmo inovador proposto por Kirkpatrick, C. D. Gelatt e Vecchi (1983). O seu funcionamento foi inspirado no processo

estocástico de equilíbrio térmico proposto por Metropolis *et al.* (1953), tendo sido aplicado inicialmente para resolver um famoso problema combinatório, o *problema do caixeiro viajante*. Este problema consiste em encontrar a menor rota possível para que um caixeiro possa percorrer diversas cidades, visitando-as uma de cada vez até retornar à origem. Bons resultados foram obtidos por este método e, posteriormente, o algoritmo foi aplicado para problemas de otimização em diversas áreas.

Os Algoritmos Genéticos (GA) foram desenvolvidos por Holland (1975) e amplamente divulgados por Goldberg (1989) em seu livro. Este tipo de algoritmo busca referências nos campos da genética e no evolucionismo de Darwin, utilizando critérios como a *sobrevivência do mais forte*, a *reprodução* e a *mutação de genes*. Os Algoritmos Genéticos introduziram nos metaheurísticos procedimentos importantes que, posteriormente, foram utilizados na criação de diversos outros métodos. São eles:

- O emprego de uma população de pontos a cada iteração, em vez de um ponto único
- A utilização somente do valor resultante da função objetivo, não sendo necessária nenhuma informação interna
- A ideia de gerações (iterações) criadas a partir da reprodução e mutação dos indivíduos
- A introdução do conceito de sobrevivência do mais forte por meio de regras probabilísticas

Inspirados na funcionalidade de tais métodos, bem sucedidos na resolução de problemas de otimização mais robustos e, por vezes, trazendo alguns novos conceitos, um vasto número de metaheurísticos surgiram. Alguns algoritmos metaheurísticos relevantes que surgiram posteriormente são: *Ant Colony Optimization - ACO* (Otimização por colônia de formigas) por Marco Dorigo Alberto Colorni (1991), *Particle Swarm Optimization - PSO* (Otimização por Enxame de Partículas) por Kennedy e Eberhart (1995), *Harmony Search Algorithm - HS* (busca harmônica) por Geem e Kim (2001), *Firefly Algorithm - FA* (Algoritmo Vaga-lume) por Yang (2009) e *Enhanced Collision Bodies Optimization - ECBO* (Otimização por Colisão de Corpos Melhorada) por Kaveh e Ghazaan (2014).

2.2 OTIMIZAÇÃO NA ENGENHARIA CIVIL

Após o surgimento dos métodos metaheurísticos (métodos avançados de otimização), pesquisadores do campo da engenharia civil passaram a aplicá-los para resolver uma grande variedade de problemas. Isso levou a um conhecimento cada vez maior sobre o funcionamento desses métodos, bem como a uma nova visão de como os computadores poderiam auxiliar o trabalho dos engenheiros. De acordo com Kaveh (2017), já que os recursos, financiamento e tempo são sempre limitados nos projetos de engenharia, existe

um forte apelo para a utilização de técnicas que possam obter soluções menos dispendiosas nesses quesitos.

Segundo Dede1 *et al.* (2019), entre as diferentes áreas da engenharia civil, as que obtiveram os melhores resultados com otimização por meio de metaheurísticos foram as seguintes:

- Geotecnia
- Transportes
- Gestão da Construção
- Hidráulica
- Estruturas

No campo da **geotecnia**, diversos trabalhos recentes foram publicados utilizando metaheurísticos para otimização. As principais aplicações deram-se em muros de contenção de terra, predição da estabilidade de taludes e paredes de cisalhamento.

Yepes *et al.* (2008) apresentaram um estudo sobre a otimização de muros de contenção de terra que utilizava o *Simulated Annealing* para otimizar muros entre 4m e 10m de altura para diferentes disposições de preenchimento e geometria. O custo foi utilizado como função objetivo e as variáveis de projeto eram propriedades geométricas e tipos de material e reforço aplicados. Já Gandomi, Roke e Mousavi (2015) estudaram diferentes técnicas de otimização, como Accelerated Particle Swarm Optimization (APSO), Firefly Algorithm (FA) e Cuckoo Search (CS) para otimização de muros de contenção tipo *cantilever*. Os autores basearam-se no procedimento ACI 318-05, utilizando-se de variáveis contínuas para a geometria do muro e variáveis discretas para o reforço de aço. No trabalho, foram utilizados tanto o peso quanto o custo do muro como funções objetivo da otimização.

A área de **transportes**, tão importante para o desenvolvimento da civilização, mostrou-se um campo fértil para o emprego de técnicas de otimização. Isto se explica dada a variedade de aplicações encontradas nessa área, incluindo planejamento, projeto, construção e operação de rodovias, tráfego, infraestruturas, ferrovias, portos e aeroportos. Uma aplicação importante das técnicas de otimização é na simulação de tráfego em rodovias, em que um grande número de variáveis afeta a qualidade de uma rede. Além disso, recentemente, conceitos de sustentabilidade econômica, social e ambiental têm sido empregados na área.

Liu e Chang (2011) estudaram a otimização de um problema de sinal arterial por meio de uma solução de algoritmo genético (GA). Ao final da otimização, duas possíveis soluções poderiam ser escolhidas de acordo com as funções objetivo: minimizar o tempo de viagem total ou maximizar o rendimento total sobre a área de destino. O algoritmo foi aplicado para um exemplo arterial de quatro interseções, para diferentes cenários de

demanda. Segundo os autores, o modelo proposto é promissor para aplicação em sinais arteriais, sobretudo em tráfegos congestionados e com alta demanda. Sharma e Kumari (2015) apresentaram uma revisão de literatura sobre a utilização dos algoritmos Ant Colony Optimization (ACO), Bacterial Foraging Optimization (BFO) e Particle Swarm Optimization (PSO) na otimização de tráfego. O estudo revelou que o PSO reduziu significativamente o tráfego nos problemas analisados, além de reduzir a complexidade e o tempo de convergência no processo de otimização quando comparado aos outros métodos. O autor também concluiu que a utilização do ACO é uma abordagem extremamente viável para solucionar problemas sequenciais em redes ad-hoc veiculares (VANETs).

A área de **gestão e planejamento da construção** também ganhou seu espaço na otimização por metaheurísticos. Essa área envolve planejamento de obra, gestão de contrato, gerenciamento do processo de construção, controle de estoque e controle de custos. Os processos estabelecidos nesse setor são complexos e, muitas vezes, subjetivos quando comparados às outras áreas da engenharia civil citadas. Isso ocorre pois áreas como transportes, estruturas e geotecnia sempre se desenvolveram em torno de modelos matemáticos objetivos, diferentemente do campo de gestão de obras, que é muito afetado pelos aspectos gerenciais e interpessoais dos seus processos. Ainda assim, devido à flexibilidade dos metaheurísticos diante das propriedades das funções objetivo, a otimização passou recentemente a ser implementada na área, com bons resultados. Normalmente, a otimização em gestão de obras busca valores ótimos em tempo, custo, qualidade, ambiente e segurança.

Azeez e Alsaffar (2014) utilizaram o algoritmo Ant Colony Optimization (ACO) para uma otimização multi-objetivo envolvendo tempo e custo. O objetivo era encontrar soluções com um balanceamento entre os dois objetivos (tempo e custo) dentro do problema de gestão de um projeto de construção. Para a criação do modelo, sete atividades de obra foram selecionadas, sendo que cada atividade poderia conter métodos diferentes (variações da atividade com características diferentes), o que resultou em 4.860 possíveis caminhos (soluções). Como resultado, o tempo foi reduzido em 54,4% e o custo reduzido em 15%. De forma similar, Elbeltagi *et al.* (2016) utilizaram tempo, custo, recursos e fluxo de caixa para otimizar um problema de agendamento de atividades de um projeto. Foi utilizado o algoritmo Particle Swarm Optimization (PSO) para otimizar um projeto dividido em 24 atividades principais. Bons resultados foram obtidos com esse trabalho.

No campo da **hidráulica**, os principais problemas encontrados envolvem a expansão de sistemas de distribuição de água, identificação da fonte de poluição de águas subterrâneas, redução de vazamentos, estimativa de consumo de energia, projeto de redes hidráulicas, entre outros. Tendo em vista esses diferentes problemas, muitos pesquisadores começaram a testar técnicas de otimização, utilizando, em grande parte, algoritmos metaheurísticos.

Fayad, Peralta e Forghani (2012) estudaram a otimização do uso conjuntivo de

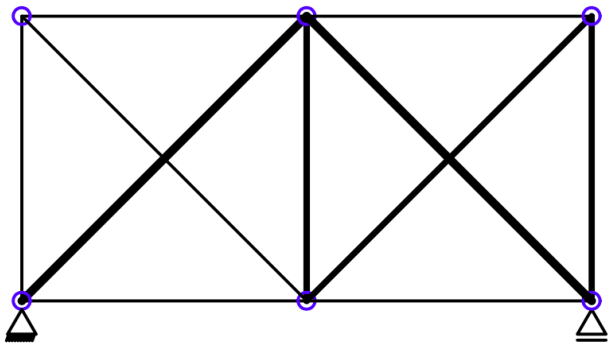
água através de um modelo que utiliza Redes Neurais Artificiais (ANN). Em seu trabalho, uma otimização multi-objetivo foi utilizada, sendo o primeiro a maximização da média de água fornecida por diferentes fontes e o segundo a maximização da energia produzida por hidrelétricas, potencializando a liberação do reservatório pela turbina. Já Bayram *et al.* (2015) avaliaram, pela primeira vez, a performance do algoritmo Teaching-learning Based Optimization (TLBO) na estimativa da concentração de oxigênio dissolvido na rede. Os resultados obtidos foram comparados com alguns outros métodos tradicionais e com o algoritmo Artificial Bee Colony (ABC). Os autores constataram que o TLBO obteve o melhor resultado na estimativa de concentração de oxigênio dissolvido.

2.3 OTIMIZAÇÃO EM ESTRUTURAS

A otimização por metaheurísticos na área de estruturas mostrou-se uma das mais aplicadas no campo da engenharia civil nas últimas décadas. De fato, um grande número de artigos foram publicados envolvendo os mais diferentes tipos de estruturas, sejam pontes, pórticos, treliças planas e espaciais, coberturas metálicas, torres estaiadas, torres de linha de transmissão e muitos outros. Um dos motivos da grande aplicabilidade destes métodos na engenharia estrutural está ligado à maturidade da área, oferecendo modelos matemáticos muito bem validados para a criação de funções objetivo, variáveis e restrições claras e objetivas.

Segundo Harsono *et al.* (2020), em se tratando de estruturas, existem três possíveis tipos de otimização: **dimensional** [*size*], **geométrica** [*shape*] e **topológica** [*topology*]. A otimização **dimensional** ocorre pela mudança nas seções transversais dos elementos estruturais. Como demonstrado na Figura 4, a mudança nas seções de uma treliça é considerada uma otimização dimensional, e esta pode ocorrer por meio de seções de valor contínuo ou discreto (por exemplo, para considerar apenas uma lista de seções metálicas existentes no mercado local).

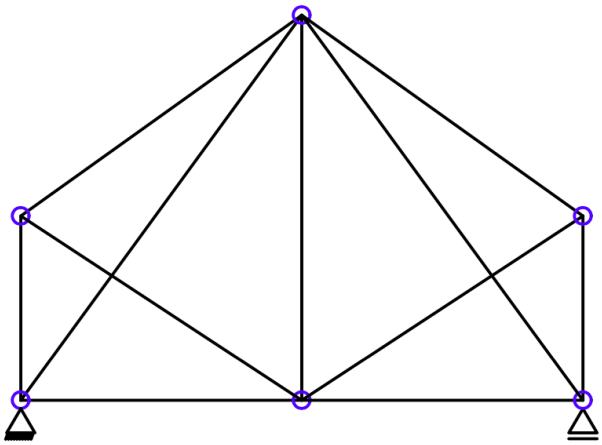
Figura 4 – Exemplo de otimização dimensional de treliças



Fonte: Autor

Já a otimização **geométrica** altera o formato da estrutura ao alterar as posições dos seus nós. A Figura 5 demonstra, para o mesmo caso anterior, um exemplo de otimização geométrica.

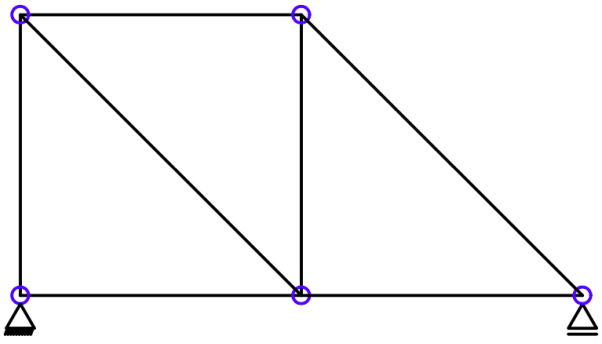
Figura 5 – Exemplo de otimização geométrica de treliças



Fonte: Autor

Por fim, a otimização **topológica** é aquela que altera o número de elementos de uma estrutura (exclui ou adiciona) para encontrar a solução ótima. A Figura 6 demonstra esse tipo de otimização em treliças.

Figura 6 – Exemplo de otimização topológica de treliças



Fonte: Autor

De acordo com Vanderplaats (1992), a primeira aplicação computacional de otimização estrutural foi feita por Schmit em 1960, ao otimizar uma treliça plana de três nós. A partir daí, sobretudo nas últimas duas décadas, diversas pesquisas foram feitas na área.

De forma mais recente, Souza, Miguel, Lopez *et al.* (2016) utilizaram o Firefly Algorithm (FA) de Yang (2009) e o Backtracking Search Algorithm (BSA) de Civicioglu (2013) para otimizar torres de linha de transmissão. Ao dividir a estrutura em diferentes

módulos, com topologias pré-definidas, foi desenvolvido um procedimento para a otimização topológica, geométrica e dimensional desse tipo de estrutura. Para os dois tipos de problema testados, o método conseguiu reduzir em até 6,4% o peso das torres de transmissão, quando comparado a métodos clássicos de otimização.

O algoritmo Firefly (FA) também foi utilizado na área estrutural por outros dois autores. Silveira (2014) utilizou o Firefly para a otimização de três problemas de otimização em cima de pórticos planos. No estudo, foram consideradas restrições relacionadas a geometria da seção transversal além de tensões devido a esforços internos da estrutura. Ribeiro (2014) utilizou o algoritmo para a otimização dimensional e geométrica do peso de treliças planas, aplicando-o a três exemplos e comparando seus resultados com outros autores da literatura.

Barbaresco (2014) utilizou o algoritmo de competição imperialista (ICA) para a otimização de uma variedade de problemas de engenharia, sobretudo na otimização dimensional e geométrica do peso de treliças planas. Bons resultados foram encontrados ao realizar comparações estatísticas do modelo, o qual obteve mínimos bastante próximos aos outros autores.

Outros quatro autores estudaram recentemente o Search Group Algorithm - SGA (algoritmo aplicado neste estudo) para a otimização estrutural. Carraro (2015) estudou a aplicação do SGA na otimização de pórticos planos com uma análise linear, obtendo alguns dos melhores resultados entre os autores comparados. Segundo o autor, o algoritmo obteve bons resultados para todos os três problemas aplicados. De forma similar, Yamamoto (2015) utilizou o SGA para otimização de pórticos planos, porém com uma análise não-linear. Os resultados foram comparados aos obtidos por Carraro (2015), onde a análise mais rigorosa utilizada (de segunda ordem) permitiu identificar falhas em alguns dos pórticos otimizados por uma análise linear. Uma abordagem diferente foi estudada por Roman (2016), em que foi realizada a otimização de uma cobertura em aço de um edifício industrial utilizando o SGA. Foi aplicada a otimização simultânea dimensional, geométrica e topológica da estrutura. Já Furtado (2018) utilizou o SGA para a otimização dimensional de longarinas de concreto em pontes. Considerando as normas ABNT, o autor desenvolveu um algoritmo de cálculo estrutural pelo método de Fauchart para otimizar longarinas com diferentes configurações de vãos. O objetivo da otimização era a redução do custo (tanto de material, quanto de mão de obra) e, segundo o autor, foram obtidos resultados satisfatórios.

Entre as diferentes abordagens da otimização no campo das estruturas, a otimização de treliças mostrou-se interessante para avaliar a performance de algoritmos metaheurísticos. De forma ideal, treliças são estruturas reticuladas cujas barras possuem todas as suas extremidades rotuladas e com cargas aplicadas somente em seus nós. Portanto, a conformação utilizada nas treliças garante uma maior liberdade na alteração da geometria da estrutura, sendo facilmente utilizada para otimização conjunta de dimensão, geometria

e topologia.

Devido a essa característica das treliças, diversos algoritmos metaheurísticos de otimização foram comparados utilizando problemas padronizados de otimização de treliças planas e espaciais. Em geral, esses problemas utilizam o peso ou a frequência natural como função objetivo e possuem variações a respeito das restrições utilizadas, como flambagem, tensão admissível, máximo deslocamento dos nós, entre outros.

A utilização de otimização dimensional e geométrica em treliças é bastante comum na literatura, porém a adição da otimização topológica é bem mais recente e exige a utilização de algoritmos mais robustos. Sobre esse tipo de otimização multimodal, pode-se citar Miguel, Lopez e Miguel (2013), que utilizaram o algoritmo Firefly (FA) para a otimização do peso da estrutura em problemas padrão de treliças planas e espaciais, comparando seus resultados com os de outros autores. O algoritmo Firefly, aplicado nesse estudo, obteve os melhores mínimos globais em quase todos os exemplos testados. Ainda, Harsono *et al.* (2020) estudaram problemas de treliça semelhantes, fazendo comparações entre a performance de três variantes do algoritmo Particle Swarm Optimization (PSO). Nesse estudo, treliças planas foram submetidas a restrições de estabilidade, tensão e deslocamento. A variante BBPSO do algoritmo foi a que obteve os menores valores para o peso das estruturas testadas, sendo definida como a melhor variante pelo autor.

De forma similar aos estudos supracitados, o presente trabalho fará comparações da performance do Search Group Algorithm (SGA) com outros resultados da literatura, ao otimizar estruturas treliçadas de forma topológica, geométrica e dimensional.

3 SEARCH GROUP ALGORITHM - SGA

O Search Group Algorithm (SGA) é um algoritmo metaheurístico desenvolvido por Gonçalves, Lopez e Miguel (2015) tendo em vista a aplicação em problemas de otimização estrutural de treliças. O algoritmo baseia-se na escolha de grupos de busca que, a cada iteração, geram novos indivíduos de uma família. O seu funcionamento dinâmico garante uma boa performance ao buscar soluções em problemas robustos, como a aplicação simultânea de otimização dimensional, geométrica e topológica (SSTO) e a utilização de variáveis discretas em problemas estruturais.

Segundo Gonçalves, Lopez e Miguel (2015), os pontos que fazem o Search Group Algorithm diferente de outros algoritmos são:

- A estratégia de que, quanto melhor o grupo de busca, mais indivíduos ele gera em determinada iteração;
- A estratégia de mutação baseada no valor médio e desvio padrão da posição do atual membro do grupo de busca em determinada iteração;
- O esquema de seleção do próximo grupo de busca, que inclui uma (i) fase global, em que o novo grupo de busca é formado pelo melhor indivíduo de cada família e uma (ii) fase local, em que o novo grupo de busca é formado pelos melhores indivíduos entre todas as famílias.

Sendo assim, a escolha do SGA neste trabalho foi baseada na performance do algoritmo ao tratar problemas robustos de otimização em treliças, além da capacidade de adaptação dos parâmetros para refinar os resultados obtidos (ajustar busca local e global, por exemplo).

O algoritmo de otimização utilizado no presente trabalho foi desenvolvido por André Jinklings[1] na linguagem de programação ***Python*** e modificado neste estudo. Todo o funcionamento do SGA é explicado nas seções seguintes.

3.1 GERAÇÃO DA POPULAÇÃO INICIAL

Por ser um algoritmo evolutivo, o SGA trabalha, a cada iteração, com uma população, cujo número de indivíduos é definido como parâmetro previamente. Cada indivíduo da população é descrito por uma combinação das variáveis de projeto e pelo valor da sua função objetivo.

O primeiro passo do Search Group Algorithm é gerar a população inicial, isto é, o conjunto inicial de indivíduos a serem testados pela função objetivo na primeira iteração.

[1] Base para o algoritmo de otimização deste estudo desenvolvido por André Jinklings, disponível em: <https://github.com/Ginklings/pysga>. Acesso em 28 de fev. de 2022.

O SGA faz a geração desses indivíduos de forma aleatória, respeitando os limites superior e inferior de cada variável de projeto (dentro do domínio), como na equação (2):

$$P_{ij} = X_j^{min} + (X_j^{max} - X_j^{min})U[0,1], \quad j = 1 \text{ até } n, \; i = 1 \text{ até } n_{pop} \tag{2}$$

Onde n_{pop} é o tamanho da população, n é a quantidade de variáveis, X_j^{min} é o limite inferior da variável, X_j^{max} é o limite superior da variável e $U[0,1]$ é uma variável aleatória uniforme entre 0 e 1.

Assim, para cada indivíduo, as n variáveis são geradas de forma aleatória dentro do domínio, formando a população inicial com n_{pop} indivíduos.

3.2 SELEÇÃO DO GRUPO DE BUSCA INICIAL

Após a geração da população inicial, todos os indivíduos são avaliados a partir da função objetivo e ordenados conforme sua colocação. Dessa população, um grupo de elite contendo os melhores indivíduos é selecionado diretamente para o grupo de busca.

O resto dos indivíduos são selecionados por um método de torneio. Esse método separa os indivíduos restantes em grupos de forma aleatória. O torneio seleciona o melhor indivíduo de cada grupo para fazer parte do grupo de busca inicial.

É importante reforçar que a escolha do torneio nesta etapa permite que bons indivíduos tenham maior chance de serem escolhidos para o grupo de busca, porém não a garantem sua escolha. Dessa forma, a otimização inicia com uma maior diversidade de indivíduos, sendo mais eficiente ao lidar com problemas de muitos mínimos locais.

3.3 MUTAÇÃO DO GRUPO DE BUSCA

Nos algoritmos evolutivos, a mutação é uma ferramenta que permite diversificar a população ao modificar o valor das variáveis de alguns indivíduos.

No SGA, a mutação de indivíduos do grupo de busca é aplicada tanto na fase global, quanto na fase local de otimização. O método para realizar a mutação é baseado nos valores de média e desvio padrão de cada variável. A ideia é afastar a variável de valores comuns no grupo de busca, fazendo o algoritmo explorar novos domínios. A equação de mutação é descrita a seguir:

$$X_j^{mutation} = \bar{x_j} + t\epsilon\sigma, \quad j = 1 \text{ até } n \tag{3}$$

onde $\bar{x_j}$ é a média e σ é o desvio padrão da variável j dentro dos indivíduos grupo de busca, ϵ é um valor aleatório conveniente e t é um valor que controla a variação de cada indivíduo mutado. Normalmente, assume-se um valor inteiro de t diferente para cada indivíduo mutado partindo de 1 (por exemplo, para 5 indivíduos mutados t=1, 2, 3, 4, 5). A partir do valor de t descrito acima, é possível compreender como a mutação pode ser um passo importante para a diversificação da população.

A escolha dos indivíduos a serem mutados é realizada a partir de um método de torneio reverso, que é semelhante ao método abordado anteriormente, em que o grupo de busca é separado em subgrupos de forma aleatória. A diferença deste método é a escolha do pior indivíduo de cada grupo (máximo valor da função objetivo), além de não haver, nesta etapa, a escolha de um grupo de elite.

A mutação do grupo de busca ocorre em todas as iterações do algoritmo, logo após a escolha do grupo de busca. Já $n_{perturbed}$ (número de indivíduos mutados) é um parâmetro do SGA que pode ser alterado previamente.

3.4 GERAÇÃO DAS FAMÍLIAS A PARTIR DE CADA GRUPO DE BUSCA

No SGA, cada integrante do grupo de busca gera outros indivíduos, formando uma família. Dessa forma, existem tantas famílias quantos são os indivíduos do grupo de busca.

Uma das características marcantes do Search Group Algorithm é que a quantidade de descendentes gerada por cada integrante do grupo de busca é determinada pelo seu *ranking*. Assim, bons indivíduos geram mais descendentes, permitindo uma melhor exploração da função objetivo nessa região do domínio.

3.4.1 Quantidade de descendentes gerados

A definição da quantidade de descendentes gerados por cada indivíduo do grupo de busca pode seguir qualquer função arbitrada pelo usuário (GONÇALVES; LOPEZ; MIGUEL, 2015).

No caso do algoritmo implementado neste trabalho, a regra utilizada calcula um número de descendentes específico para cada indivíduo do grupo de busca, baseado no seu *ranking*. Essa regra pode ser demonstrada através de um exemplo.

Inicialmente, é gerada uma matriz de vetores de tamanho $n_g + 1$, seguindo a seguinte equação:

$$x_i = (n_{pop} - n_g)\left(1 - \frac{i-1}{n_g}\right)^2, \quad \text{i} = 1 \text{ até } n_g + 1 \tag{4}$$

No caso de um grupo de busca de tamanho $n_g = 10$ e população $n_{pop} = 80$, a matriz gerada é:

$$X_{descendentes} = [70 \;\; 56.7 \;\; 44.8 \;\; 34.3 \;\; 25.2 \;\; 17.5 \;\; 11.2 \;\; 6.3 \;\; 2.8 \;\; 0.7 \;\; 0]$$

Após, cada elemento é substituído por $x_i - x_{i+1}$ (elemento atual menos elemento seguinte, reduzindo o tamanho da matriz para n_g). É importante ressaltar que esse valor é arredondado para o inteiro mais próximo, não podendo ser menor que 1:

$$X_{descendentes} = [13 \;\; 12 \;\; 10 \;\; 9 \;\; 8 \;\; 6 \;\; 5 \;\; 4 \;\; 2 \;\; 1]$$

Por fim, para que a soma total dos valores dessa matriz sejam equivalentes ao número de indivíduos a serem gerados, o valor do primeiro elemento é subtraído por $\sum(X_{descendentes}) - n_{pop} + n_g$ (neste caso, subtrai-se 1):

$$X_{descendentes} = [12 \;\; 12 \;\; 10 \;\; 9 \;\; 8 \;\; 6 \;\; 5 \;\; 4 \;\; 2 \;\; 1]$$

A leitura da matriz resultante é feita com relação às posições i= 1 até n_g e representa o número de descendentes a serem gerados por aquele indivíduo. Assim, por exemplo, o indivíduo de ranking 4 no grupo de busca geraria 9 novos indivíduos.

3.4.2 Geração de novos indivíduos na fase global

A geração de descendentes a partir do grupo de busca é realizada a partir de uma perturbação α, calculado para cada variável de projeto. O α é um índice de mesma ordem de grandeza da variável a ser alterada.

A função que gera os descendentes no SGA é diferente para as fases local e global. Ambas utilizam a perturbação α, porém as equações que o definem são diferentes. Para a fase global, α é calculado pela seguinte equação:

$$\alpha = (\alpha_0 \beta + \alpha_{min})(l_{sup} - l_{inf}) \tag{5}$$

onde α_0 (alfa inicial) e α_{min} (alfa mínimo) são parâmetros constantes definidos previamente, β é o fator de decaimento e l_{sup} e l_{inf} são, respectivamente, os limites superior e inferior da variável a ser perturbada.

O fator de decaimento β possui valores entre 0 e 1 e é responsável por reduzir a perturbação a cada iteração na fase global. O intuito dessa redução é permitir que o algoritmo explore o domínio do problema de forma ampla no início da otimização, com uma alta aleatoriedade, mas que, ao longo das iterações, comece a se restringir às regiões de busca que se mostraram mais promissoras.

O fator β é calculado por meio da seguinte equação:

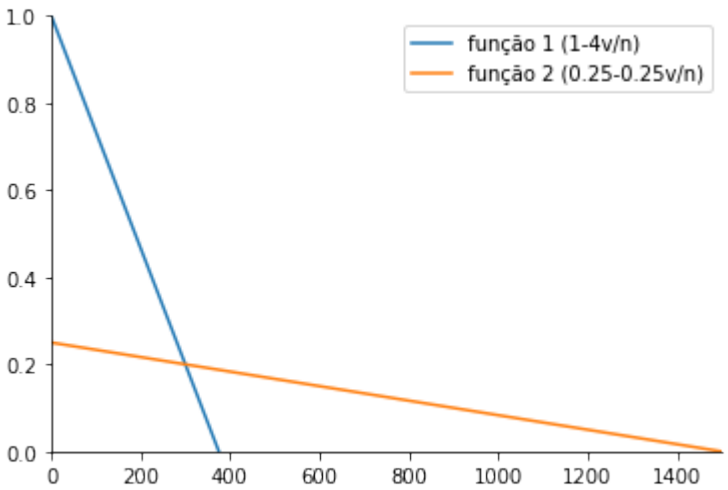
$$\beta = max \begin{cases} 1 - \frac{4\,v}{n_{iter}} \\ 0.25 - \frac{0.25\,v}{n_{iter}} \end{cases} \tag{6}$$

onde v é a iteração atual e n_{iter} é o número total de iterações globais.

Como se observa, β recebe o valor máximo entre as duas equações. Graficamente, cada uma dessas equações representa uma reta decrescente, a primeira com uma inclinação maior que a segunda.

Esse comportamento faz com que a aleatoriedade da otimização decresça rapidamente até $\beta = 0.2$ e, posteriormente, tenha uma redução suave até o fim da fase global. Esse comportamento pode ser observado na figura 7

Figura 7 – Índice β para 1500 iterações



Fonte: Adaptado de Carraro (2015)

Com o valor de α definido, um valor aleatório é escolhido dentro do intervalo $[x - \frac{\alpha}{2},\ x + \frac{\alpha}{2}]$ para cada variável do problema, gerando um novo descendente. Esse processo é realizado múltiplas vezes até se obter uma nova geração com n_{pop} indivíduos.

3.4.3 Geração de novos indivíduos na fase local

Na fase local, entende-se que o algoritmo já explorou grande parte do domínio e, assim, busca-se fazer pequenas alterações nos indivíduos para refinar as soluções já encontradas. É por este motivo que o valor da aleatoriedade α na fase global segue a seguinte equação:

$$\alpha = (\alpha_{min}\ \beta + r_{peq}\ \alpha_{min})(l_{sup} - l_{inf}) \tag{7}$$

onde r_{peq} é um valor pequeno entre 0 e 1, responsável por manter um valor residual de α_{min}, nunca zerando o valor de α. Nota-se que, na fase local, o valor de α segue uma redução do parâmetro α_{min}, ao contrário de α_0, como ocorre na fase global.

O fator de decaimento β também sofre uma mudança importante ao seguir o valor de apenas uma reta, como na equação:

$$\beta = \frac{n_{iter} - v}{n_{iter}} \tag{8}$$

3.5 SELEÇÃO DE UM NOVO GRUPO DE BUSCA

Após a geração dos indivíduos de cada família, cada novo indivíduo é avaliado pela função objetivo. É importante lembrar que o indivíduo "pai"de cada família é diretamente

alocado na mesma, sem nenhuma perturbação. O método de seleção do novo grupo de busca difere ligeiramente para as fases global e local.

Para a **fase global**, é selecionado o melhor indivíduo de cada família para compor o novo grupo de busca. Esse método é importante para a fase global, pois o algoritmo consegue manter maior diversidade da população, com cada família explorando um local do domínio.

Já para a **fase local**, são selecionados os melhores indivíduos dentre toda a população, independente da família à qual o indivíduo pertence. Em ambas as fases de otimização, o novo grupo de busca é ordenado de acordo com sua qualificação e segue para uma nova iteração.

3.6 PRINCIPAIS PARÂMETROS

A escolha dos parâmetros no Search Group Algorithm é um passo essencial para a boa performance do algoritmo na escolha de boas soluções. Em geral, cada problema possui diferentes parâmetros ótimos a serem escolhidos. É por isso que o usuário deve ter boa compreensão do funcionamento do SGA, para que possa testar diferentes parâmetros (*parameter tunning*).

Na tabela abaixo, são apresentados os principais parâmetros do SGA utilizados no presente trabalho:

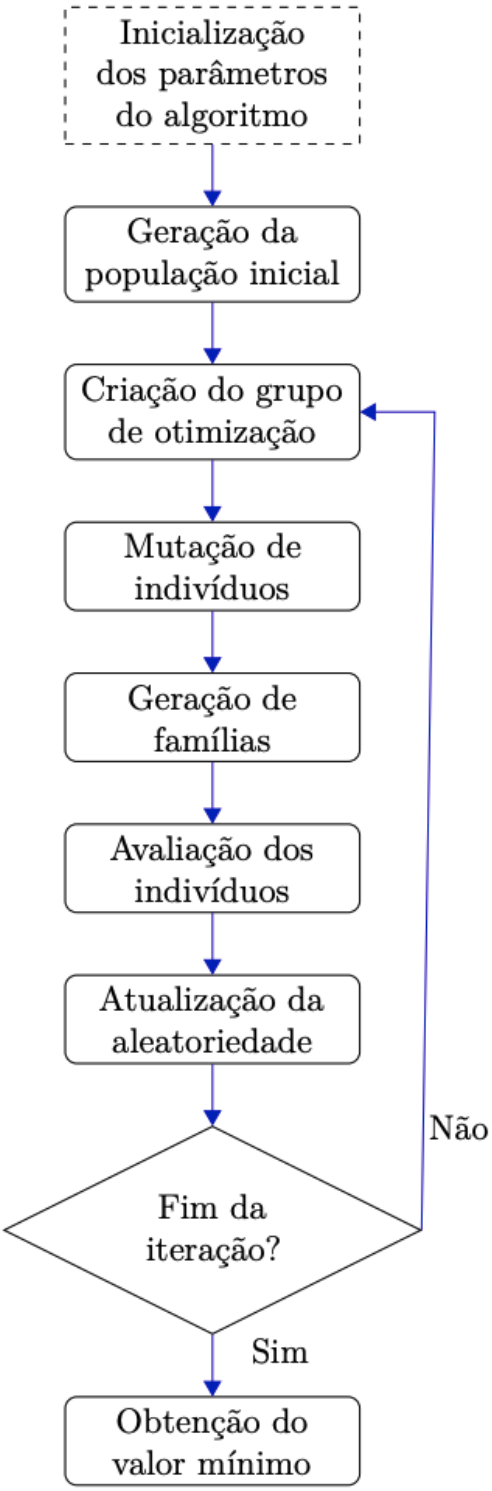
Tabela 1 – Lista de Parâmetros do SGA

Parâmetro	**Descrição**
α_{min}	Índice de perturbação mínimo, utilizado para a geração de novos indivíduos na fase local.
α_0	Índice de perturbação inicial, utilizado para a geração de novos indivíduos na fase global.
$N_{iterations}$	Número total de iterações do algoritmo.
It_{ratio}^{global}	Taxa de iterações globais. Representa a porcentagem das iterações totais dedicadas à fase global
n_{pop}	Tamanho da população. Representa o número de indivíduos testados a cada iteração do algoritmo.
SG_{ratio}	Taxa de grupo de busca. Representa a porcentagem da população que será escolhida para formação do grupo de busca.
$N_{perturbed}$	Número de indivíduos perturbados. Representa o número de indivíduos que devem ser mutados a cada iteração do algoritmo.

Fonte: Autor

3.7 FLUXOGRAMA DO ALGORITMO

Figura 8 – Rotina iterativa do algoritmo SGA



Fonte: Carraro (2015)

3.8 APLICAÇÃO EM FUNÇÃO DE TESTE

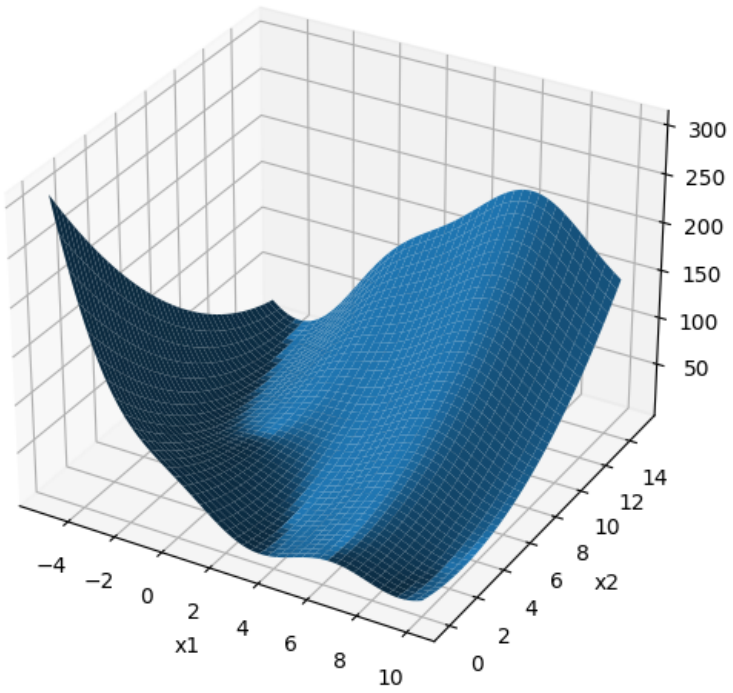
Para validar a eficácia do algoritmo SGA desenvolvido na linguagem *Python* em problemas de otimização, será utilizada uma função de teste. Funções matemáticas são comumente utilizadas para a calibração e teste de modelos de otimização, os quais normalmente possuem diferentes mínimos locais que dificultam a otimização por métodos tradicionais. A função *Branin* ou *Branin-hoo* será utilizada para esta validação.

A função Branin possui três mínimos globais com valor 0,397887 e é normalmente utilizada nos limites $x_1 \in [-5, 10], x_2 \in [0, 15]$. Essa função possui duas dimensões (x_1 e x_2) e é descrita da seguinte forma:

$$f(x) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1 - t)cos(x_1) + s \tag{9}$$

Onde $a = 1$, $b = \frac{5.1}{4*\pi^2}$, $c = \frac{5}{\pi}$, $r = 6$, $s = 10$ e $t = \frac{1}{8\pi}$. A função tridimensional está representada na figura 9.

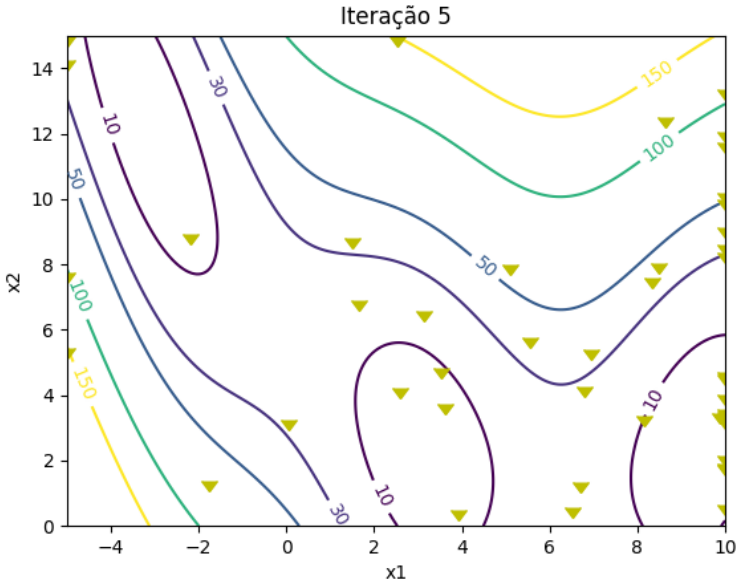
Figura 9 – Representação tridimensional da função Branin



Fonte: Autor

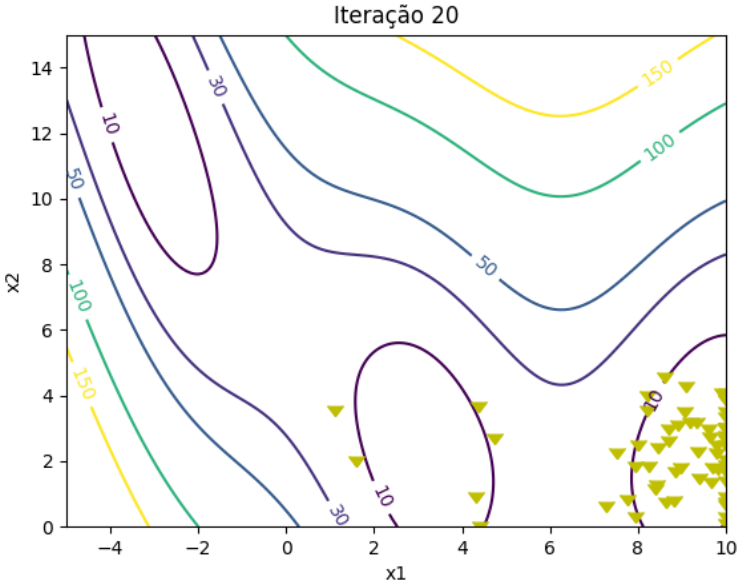
Durante o teste, utilizou-se o conjunto de parâmetros $\alpha_{min} = 0.01$, $\alpha_0 = 2$, $N_{iterations} = 100$, $It_{ratio}^{global} = 0.4$, $n_{pop} = 100$, $SG_{ratio} = 0.2$ e $N_{perturbed} = 4$. Nas figuras 10, 11, 12 e 13, demonstra-se a dispersão da população para diferentes iterações do processo de otimização. Em amarelo, as populações na fase global, e, em azul, as populações na fase local.

Figura 10 – Otimização da função Branin com SGA - Iteração 5



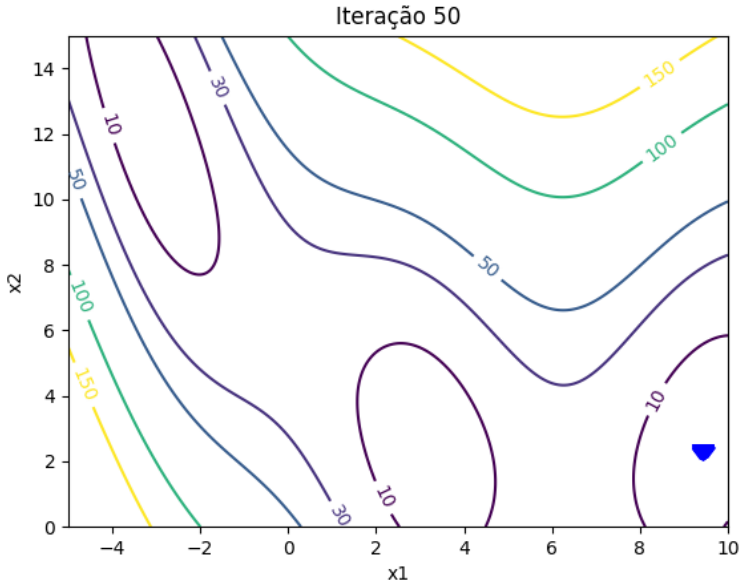
Fonte: Autor

Figura 11 – Otimização da função Branin com SGA - Iteração 20



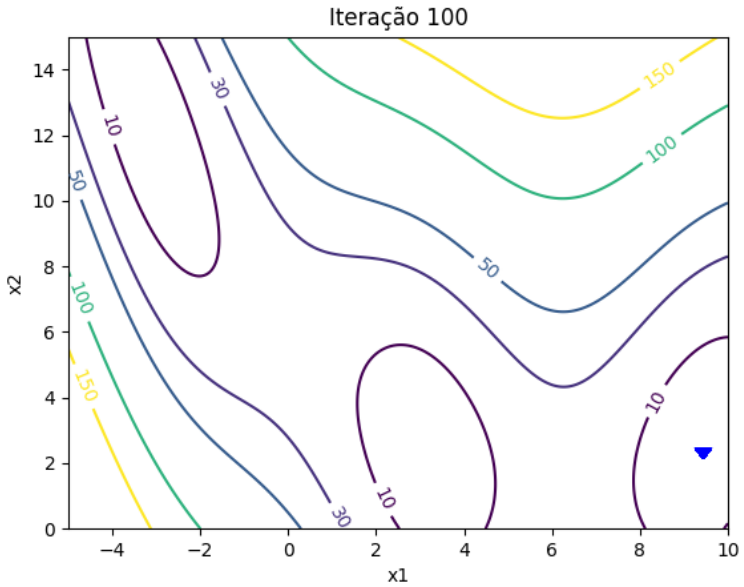
Fonte: Autor

Figura 12 – Otimização da função Branin com SGA - Iteração 50



Fonte: Autor

Figura 13 – Otimização da função Branin com SGA - Iteração 100



Fonte: Autor

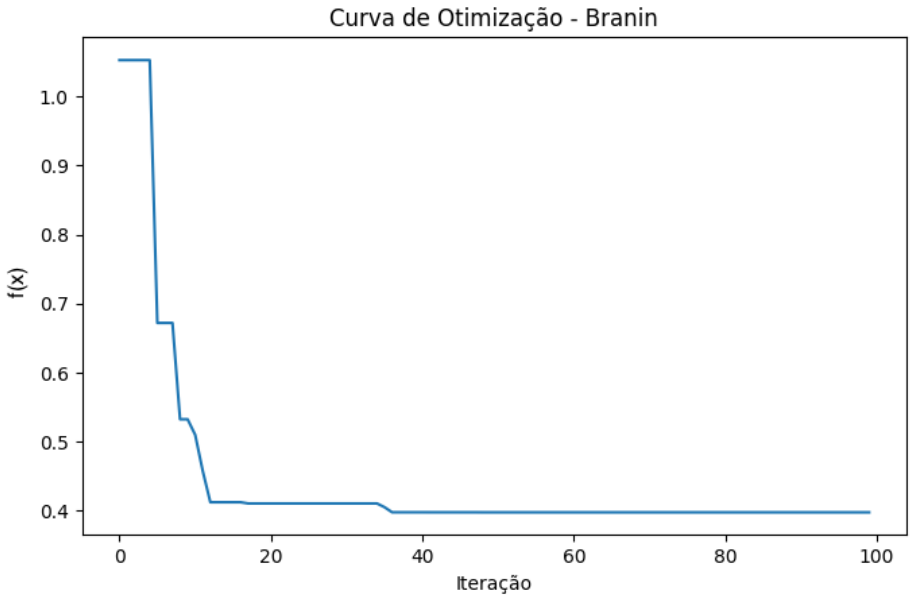
Pode-se observar que, nas iterações de busca global (amarelo), a população possui uma diversidade muito maior, realizando a exploração do domínio existente no problema. Já na fase de busca local, a região de exploração é muito menor e se concentra em um local promissor do domínio de busca. Na iteração 100, quase toda a população encontra-se

em uma região bastante específica, garantindo que o refino da solução foi realizado.

Quando analisamos o resultado obtido de $y = 0,3978874584864496$ para $x_1 = 9,42487854$ e $x_2 = 2,47531329$ e o comparamos com o mínimo real da função de 0,397887 (consideradas as casas decimais), o algoritmo conseguiu convergir para o ponto ótimo com 100% de assertividade.

Uma outra análise que pode ser feita é a construção de uma curva de otimização que descreve o mínimo encontrado em cada iteração do processo de otimização. Na curva demonstrada na figura 14, pode-se observar que o valor da função objetivo decresce rapidamente nas iterações iniciais e, a partir da iteração 40 (quando inicia-se a busca local), a variação do mínimo atingido é imperceptível. A distribuição dessa curva garante que os processos de busca global e local funcionam como o esperado.

Figura 14 – Curva de otimização pelo SGA da função Branin



Fonte: Autor

4 ANÁLISE ESTRUTURAL

O método da matriz de rigidez direta é um método de análise estrutural baseado em montar e inverter uma matriz que contém as propriedades de uma estrutura reticulada. Esse método tornou-se especialmente conhecido com a criação de computadores digitais. Weaver e Gere (1990) observam que a abordagem por matrizes oferece uma forma eficiente para descrever diversos passos da análise estrutural e pode ser facilmente programada em computadores. O uso de matrizes para realizar cálculos é bastante natural em computadores, pois elas permitem manipular grandes quantidades de números de forma simples e efetiva. Até mesmo métodos utilizados para análises de estruturas mais robustas, como os elementos finitos, são baseados em uma matriz de rigidez. É por isso que a maior parte dos softwares de cálculo estrutural foram desenvolvidos por meio desses métodos matriciais.

Assim, é fácil compreender a escolha metodológica por matrizes de rigidez em um problema de otimização, em que uma estrutura deve ser analisada de centenas a milhares de vezes. Neste estudo, a análise de estruturas mais simples com treliças planas permite que seja utilizado um método igualmente simples e eficiente: o método da matriz de rigidez direta. A análise utilizada será a **análise elástica de 1ª ordem** de treliças planas.

Para a aplicação prática dos exemplos de treliças neste trabalho, foi desenvolvido um algoritmo na linguagem *Python* que utiliza o método da matriz de rigidez direta e extrai resultados de deslocamento dos nós, forças axiais nos elementos e o cálculo do peso da estrutura (função objetivo). O método de análise e o processo de implementação do algoritmo serão discutidos nas seções seguintes.

4.1 TRELIÇAS PLANAS

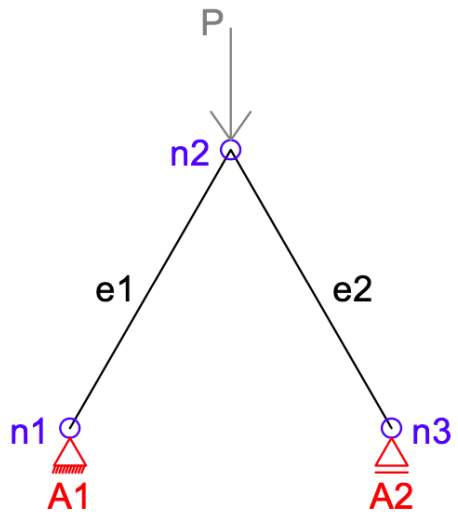
Embora o método da matriz de rigidez direta possa ser utilizado para a avaliação de diversas estruturas, como vigas, treliças espaciais, pórticos, grelhas e muitos outros, este trabalho envolve problemas de otimização em treliças planas. Portanto, o algoritmo de análise estrutural foi desenvolvido estritamente para este tipo de estrutura.

De acordo com Weaver e Gere (1990), uma treliça plana é um sistema idealizado de elementos sobre um plano, interconectados em nós rotulados. Assume-se que todas as cargas são aplicadas no plano. As cargas podem ser aplicadas nos nós ou nos elementos, porém, durante as análises, podem ser substituídas por cargas concentradas (estaticamente equivalentes) aplicadas em seus nós. Assim, ao analisar uma treliça sujeita apenas a cargas nodais, somente forças axiais de tensão e compressão serão geradas nos membros. Nos exemplos deste estudo, apenas cargas nodais serão aplicadas nas treliças, de maneira que momentos fletores e esforços cortantes podem ser desconsiderados.

4.2 COMPONENTES ESTRUTURAIS

O modelo idealizado para análise estrutural no método adotado é uma forma de aproximar a geometria e o comportamento da estrutura, dividindo-a em alguns componentes. Como exemplo, tem-se o modelo de treliça abaixo:

Figura 15 – Treliça exemplo com 2 barras



Fonte: Autor

Em vermelho estão dois apoios que restringem o movimento da estrutura, contrapondo os esforços solicitantes. No modelo ideal, os apoios não sofrem deslocamento. Pode-se observar, na figura, a simbologia de A1 como sendo um apoio de segundo gênero (garante restrição x e y) e A2 como sendo um apoio de primeiro gênero (garante restrição apenas em y).

Em azul estão representados os nós ou conexões que ligam os membros adjacentes. No caso de treliças, todos os nós são rotulados, isto é, permitem o livre giro e, portanto, não transmitem momentos fletores entre membros. Essa condição garante que treliças ideais possuam apenas esforços axiais (tração e compressão).

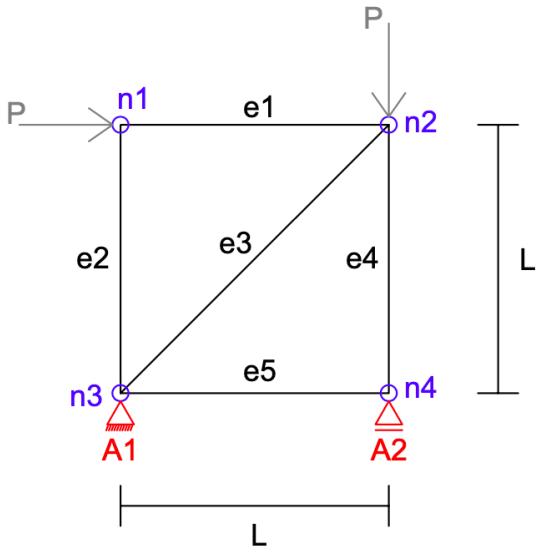
As forças ou cargas são aplicadas sempre aos nós das treliças e de forma pontual, representados no exemplo pela força vertical P, em cinza. Por fim, os elementos (ou barras) são os membros da estrutura, responsáveis pela transmissão dos esforços ao longo dos nós da estrutura.

4.3 MÉTODO DA MATRIZ DE RIGIDEZ

A explicação do método da matriz de rigidez será feita a partir da sua aplicação em uma treliça genérica biapoiada de cinco barras, como na Figura 16. Nela, as barras verticais e horizontais possuem comprimento de L e o elemento 3 possui comprimento de

L$\sqrt{2}$ (diagonal). São aplicadas forças P de igual intensidade nos nós 1 e 2. Além disso, todas as barras possuem seção transversal de área A e módulo de elasticidade E.

Figura 16 – Treliça exemplo com cinco barras



Fonte: Autor

Considerando que cada elemento de uma treliça seja representado como uma mola, fixa em 2 nós e contendo apenas esforços unidimensionais (em x), é possível representar um elemento através da equação:

$$f = k\ d \tag{10}$$

Onde f é igual força aplicada, k é relativo a rigidez da mola e d corresponde ao deslocamento.

Tendo em vista que o elemento representado conecta 2 nós, e cada nó possui 1 grau de liberdade (GDL), representamos um conjunto de equações de forma matricial:

$$\begin{bmatrix} f_1' \\ f_2' \end{bmatrix} = \begin{bmatrix} k & -k \\ -k & k \end{bmatrix} \begin{bmatrix} u_1' \\ u_2' \end{bmatrix} \tag{11}$$

Onde f_k' e u_k' correspondem à força e ao deslocamento no nó k

Para encontrar a matriz de rigidez do elemento, considera-se que um deslocamento unitário é aplicado ao nó 1, tendo $u_1' = 1$, e o nó 2 é restringido, tendo $u_2' = 0$. Dessa forma, a equação anterior é calculada:

$$\begin{bmatrix} f_1' \\ f_2' \end{bmatrix} = \begin{bmatrix} k_{11} \\ -k_{21} \end{bmatrix}$$

Assim, tendo $f_1' = k_{11}$, utiliza-se a equação de resistência dos materiais para o deslocamento de uma barra sobre carga axial para calcular o valor de k_{11}:

$$\delta = \frac{PL}{EA}$$
$$1 = \frac{k_{11}L}{EA}$$
$$k_{11} = \frac{EA}{L}$$

Da mesma forma, encontram-se todos os valores da matriz de rigidez $k_e^{'}$ para o caso de um elemento contendo esforços axiais, sendo ela:

$$k_e^{'} = \begin{bmatrix} \frac{EA}{L} & -\frac{EA}{L} \\ -\frac{EA}{L} & \frac{EA}{L} \end{bmatrix} \tag{12}$$

Considerando que todas as barras de uma treliça são axialmente carregadas e de posse dessa matriz genérica, é possível montar uma matriz de rigidez global para qualquer geometria de treliça. Entretanto, é necessário considerar que cada matriz possui um sistema de coordenadas diferente (local), já que a coordenada $x^{'}$ de cada barra tem a mesma direção de seu eixo. Ainda, na matriz de rigidez local gerada anteriormente, cada nó possui 1 GDL, portanto, uma coordenada $x^{'}$. Além disso, em uma treliça plana cada nó possui 2 GDL, com esforços nas coordenadas x e y. Sendo assim, antes de fazer a união da matriz de rigidez da estrutura, utiliza-se um método para a transformação das matrizes de rigidez locais de cada elemento para o sistema de coordenadas global da estrutura.

Para a transformação das coordenadas, considera-se um ângulo β em relação ao eixo x (horizontal) da estrutura. Assim, é estabelecida uma matriz de transformação de sistema de coordenadas:

$$\underline{\boldsymbol{T}} = \begin{bmatrix} \cos\beta & \text{sen}\beta & 0 & 0 \\ 0 & 0 & \cos\beta & \text{sen}\beta \end{bmatrix}$$

A interpretação dessa matriz pode ser feita considerando as duas linhas como a contribuição da rigidez no sistema local (unidirecional no primeiro e segundo nós) para as coordenadas globais (x e y no primeiro de segundo nós), representadas pelas quatro colunas. Por meio da utilização dessa matriz na transformação das matrizes de forças e deslocamentos, chega-se à equação de transformação da matriz de rigidez local para a global:

$$\underline{\boldsymbol{k_g}} = \underline{\boldsymbol{T}}^T \, \underline{\boldsymbol{k_e}} \, \underline{\boldsymbol{T}} \tag{13}$$

Por fim, a transformação do sistema de coordenadas local para global de um elemento de treliça plana pode ser discretizado em uma solução específica, envolvendo cossenos e senos diretores e o valor de k. Sendo:

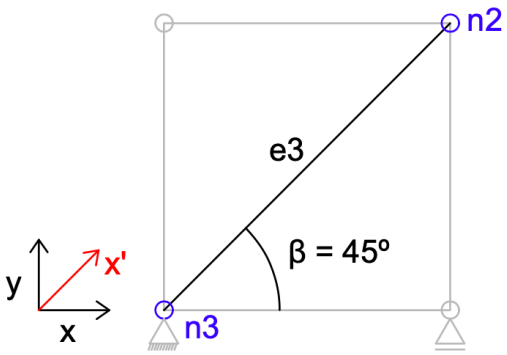
$$k_e^{'} = \begin{bmatrix} \frac{EA}{L} & -\frac{EA}{L} \\ -\frac{EA}{L} & \frac{EA}{L} \end{bmatrix} = \frac{EA}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

Então, aplicando a equação 13:

$$\underline{\boldsymbol{k}_g} = \frac{EA}{L}\begin{bmatrix} c^2 & cs & -c^2 & -cs \\ cs & s^2 & -cs & -s^2 \\ -c^2 & -cs & c^2 & cs \\ -cs & -s^2 & cs & s^2 \end{bmatrix} \tag{14}$$

Para a aplicação dessa transformação no exemplo da treliça de cinco barras para o elemento 3 (na diagonal), tem-se o valor do ângulo β. Para melhor compreensão da transformação de coordenadas utilizada, na figura abaixo é apresentada a notação x' que descreve a coordenada local da barra e3, enquanto x e y descrevem as coordenadas globais da estrutura.

Figura 17 – Detalhe do elemento 3 da treliça exemplo



Fonte: Autor

Por meio da equação 14, tem-se:

$$\underline{\boldsymbol{k}_g} = \frac{EA\sqrt{2}}{L}\begin{bmatrix} 0.27596 & 0.27596 & -0.27596 & -0.27596 \\ 0.27596 & 0.27596 & -0.27596 & -0.27596 \\ -0.27596 & -0.27596 & 0.27596 & 0.27596 \\ -0.27596 & -0.27596 & 0.27596 & 0.27596 \end{bmatrix} \tag{15}$$

É importante notar que o comprimento do elemento 3 é dado por $L\sqrt{2}$ e esse valor foi substituido na equação acima. Com os resultados da matriz rigidez do elemento 3 no sistema de coordenadas global, pode ser feita a adição dos valores obtidos na matriz de rigidez global da treliça. Para isso, é essencial compreender o significado de cada item da matriz acima. A matriz do elemento 3 foi representada na tabela abaixo, sendo que cada célula corresponde a um nó e um eixo de influência:

Assim, da mesma forma, é necessário que a adição da matriz $\underline{\boldsymbol{k}_g}$ na matriz de rigidez global da treliça $\underline{\boldsymbol{K}}$ seja feita a cada elemento, considerando a posição correta dos mesmos. Já que a treliça exemplo possui 4 nós e cada nó possui 2 GDL, a matriz $\underline{\boldsymbol{K}}$ deve ter dimensões 8 x 8. Abaixo, a adição do elemento 3 na matriz de rigidez global da treliça foi realizada e seu resultado é apresentado de forma tabular:

Tabela 2 – Representação tabular da matriz de rigidez do elemento 3

	N2 (x)	**N2 (y)**	**N3 (x)**	**N3 (y)**
N2 (x)	0.27596	0.27596	-0.27596	-0.27596
N2 (y)	0.27596	0.27596	-0.27596	-0.27596
N3 (x)	-0.27596	-0.27596	0.27596	0.27596
N3 (y)	-0.27596	-0.27596	0.27596	0.27596

Fonte: Autor

Tabela 3 – Representação tabular da matriz $\underline{\boldsymbol{K}}$ com adição do elemento 3

	N1 (x)	**N1 (y)**	**N2 (x)**	**N2 (y)**	**N3 (x)**	**N3 (y)**	**N4 (x)**	**N4 (y)**
N1 (x)	0	0	0	0	0	0	0	0
N1 (y)	0	0	0	0	0	0	0	0
N2 (x)	0	0	0.27596	0.27596	-0.27596	-0.27596	0	0
N2 (y)	0	0	0.27596	0.27596	-0.27596	-0.27596	0	0
N3 (y)	0	0	-0.27596	-0.27596	0.27596	0.27596	0	0
N3 (y)	0	0	-0.27596	-0.27596	0.27596	0.27596	0	0
N4 (y)	0	0	0	0	0	0	0	0
N4 (y)	0	0	0	0	0	0	0	0

Fonte: Autor

Observa-se que as posições relativas aos nós 1 e 4, cuja rigidez não sofre influência do elemento 3, mantiveram-se zeradas. O mesmo processo foi repetido para os elementos 1, 2 e 4, resultando na matriz de rigidez $\underline{\boldsymbol{K}}$:

$$\underline{\boldsymbol{K}} = \frac{EA}{L}\begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ -1 & 0 & 1.3535 & 0.3535 & -0.3535 & -0.3535 & 0 & 0 \\ 0 & 0 & 0.3535 & 1.3535 & -0.3535 & -0.3535 & 0 & -1 \\ 0 & 0 & -0.3535 & -0.3535 & 1.3535 & 0.3535 & -1 & 0 \\ 0 & -1 & -0.3535 & -0.3535 & 0.3535 & 1.3535 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{16}$$

Antes de utilizar a matriz de rigidez $\underline{\boldsymbol{K}}$ no cálculo de forças e deslocamentos, é necessário aplicar as restrições de apoio. Como os apoios no modelo de estrutura ideal não sofrem deslocamento ao serem submetidas a uma força, a sua rigidez é total. Assim, todas as posições na matriz correspondentes a restrições de apoio (linhas e colunas) são zeradas, com exceção da diagonal principal, em que o valor atribuído é 1. No exemplo da treliça de 5 barras, serão zeradas linhas e colunas correspondentes ao **N3 (x), N3 (y) e N4 (y)**, resultando na nova matriz $\underline{\boldsymbol{K}_g}$ final:

$$\underline{\boldsymbol{K_g}} = \frac{EA}{L}\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1.3535 & 0.3535 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.3535 & 1.3535 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{17}$$

As matrizes de deslocamento e força possuem dimensões de GDL x 1. Assim, da mesma forma que as linhas da matriz de rigidez, cada posição da matriz corresponde a um grau de liberdade de cada nó. Assim, pode-se definir a matriz de forças atuantes:

$$\underline{\boldsymbol{F}} = \begin{bmatrix} f'_{1x} \\ f'_{1y} \\ f'_{2x} \\ f'_{2y} \\ f'_{3x} \\ f'_{3y} \\ f'_{4x} \\ f'_{4y} \end{bmatrix} = \begin{bmatrix} P \\ 0 \\ 0 \\ -P \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Portanto, utilizando propriedades da geometria analítica na equação 11, pode-se encontrar o valor de $\underline{\boldsymbol{d}}$ ao multiplicar $\underline{\boldsymbol{F}}$ pela matriz inversa de $\underline{\boldsymbol{K_g}}$:

$$\underline{\boldsymbol{d}} = \underline{\boldsymbol{K_g^{-1}}}\ \underline{\boldsymbol{F}} \tag{18}$$

Com o resultado do deslocamento $\underline{\boldsymbol{d}}$, podemos aplicá-lo novamente à equação 10, utilizando $\underline{\boldsymbol{d}}$ e a matriz $\underline{\boldsymbol{K}}$ (antes da consideração das restrições de apoio) para obter uma matriz $\underline{\boldsymbol{f}}$, que contém as reações x e y em cada nó:

$$\underline{\boldsymbol{f}} = \underline{\boldsymbol{K}}\ \underline{\boldsymbol{d}} \tag{19}$$

$$\begin{bmatrix} f'_{1x} \\ f'_{1y} \\ f'_{2x} \\ f'_{2y} \\ f'_{3x} \\ f'_{3y} \\ f'_{4x} \\ f'_{4y} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ -1 & 0 & 1.3535 & 0.3535 & -0.3535 & -0.3535 & 0 & 0 \\ 0 & 0 & 0.3535 & 1.3535 & -0.3535 & -0.3535 & 0 & -1 \\ 0 & 0 & -0.3535 & -0.3535 & 1.3535 & 0.3535 & -1 & 0 \\ 0 & -1 & -0.3535 & -0.3535 & 0.3535 & 1.3535 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u'_1 \\ v'_1 \\ u'_2 \\ v'_2 \\ u'_3 \\ v'_3 \\ u'_4 \\ v'_4 \end{bmatrix}$$

Onde $u_j^{'}$ e $v_j^{'}$ são, respectivamente, o deslocamento horizontal e vertical no nó j, calculados pela equação 18.

Por fim, a matriz $\underline{\boldsymbol{f}}$, calculada na equação 19, é utilizada para a obtenção das forças e tensões atuantes em cada barra, por meio da seguinte equação:

$$for_i = \sqrt{(f_{1x}^{'} - f_{2x}^{'})^2 + (f_{1y}^{'} - f_{2y}^{'})^2} \tag{20}$$

$$ten_i = \frac{for_i}{A} \tag{21}$$

Onde for_i é a força na barra i, ten_i é a tensão na barra i, f_{1x} e $f_{2x}^{'}$ são as reações horizontais no primeiro e segundo nó da barra e $f_{1y}^{'}$ e $f_{2y}^{'}$ são as reações verticais no primeiro e segundo nó da barra.

4.4 PASSO-A-PASSO DO ALGORITMO DE ANÁLISE ESTRUTURAL

Ao longo do desenvolvimento do algoritmo de análise estrutural, algumas funcionalidades adicionais ao método da matriz de rigidez direta tiveram de ser desenvolvidas para a correta aplicação da otimização estrutural. Entre eles, destacam-se:

- O cálculo da função objetivo de peso da estrutura
- A movimentação das coordenadas dos nós da otimização geométrica
- A eliminação de elementos e nós da otimização topológica

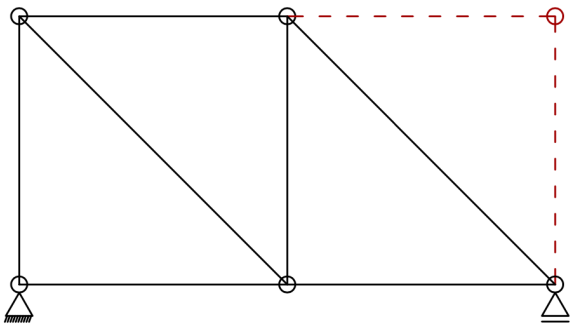
Tendo-se já o comprimento e a área da seção transversal de cada barra, a função objetivo foi calculada pela equação:

$$f(x) = \sum_{i=1}^{m} \rho_i A_i L_i \tag{22}$$

Onde ρ_i é a massa específica do material (em kg/m^2), A_i é a área da seção transversal da barra (em m^2) e L_i é o comprimento da barra (em m).

A movimentação das coordenadas geométricas e a exclusão das barras e nós foram feitas por meio da rotina do algoritmo. É válido notar que uma atenção especial deve ser dada à otimização topológica, pois a exclusão de todas as barras pertencentes ao mesmo nó exige, por consequência, a exclusão desse nó, como demonstrado na figura 18.

Figura 18 – Exclusão de nó promovido pela exclusão de barras da treliça



Fonte: Autor

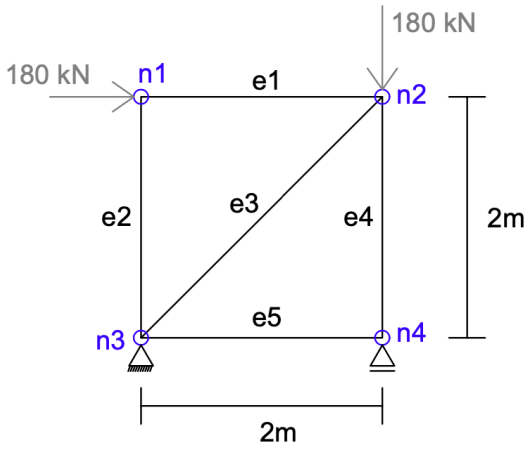
O algoritmo de análise estrutural foi particionado pela execução sequencial de diversas funções na linguagem *Python*. Assim, um pseudo-código do algoritmo pode ser generalizado da seguinte forma:

1. Definição inicial da geometria e das propriedades da treliça
2. Alteração da geometria da treliça conforme variáveis de projeto
3. Cálculo do comprimento, cosseno e seno diretores das barras
4. Criação da matriz de rigidez local
5. Transformação do sistema de coordenadas da matriz local
6. Adição da matriz de cada elemento na matriz de rigidez global
7. Consideração das condições de apoio na matriz de rigidez
8. Cálculo da matriz de deslocamentos
9. Cálculo da matriz de reações
10. Cálculo dos esforços de tensão atuantes na treliça
11. Imposição das penalizações necessárias
12. Cálculo do peso da treliça (função objetivo)

4.5 TESTE DO ALGORITMO DE ANÁLISE ESTRUTURAL

Para a validação da assertividade do algoritmo de análise estrutural desenvolvido, será feita a comparação dos resultados obtidos com o software de análise estrutural MASTAN2®. A estrutura testada será a mesma utilizada no início deste capítulo para explicar o método da matriz de rigidez.

Figura 19 – Treliça exemplo de 5 barras para validação do algoritmo



Fonte: Autor

A treliça está representada na Figura 19 e é uma estrutura biapoiada de cinco barras e quatro nós. Aplicado aos nós n1 e n2 estão duas forças de intensidade de 180 kN. Além disso, o comprimento das barras dispostas na horizontal ou vertical vale 2 metros. Outras propriedades da treliça estão descritas abaixo:

- E = 68.947591 GPa (módulo de elasticidade)
- A = 22.4 cm^2 (seção transversal para todas as barras)

A partir da execução do algoritmo de análise estrutural, a seguinte matriz rigidez foi gerada:

$$\underline{\boldsymbol{K}} = 10^7 \begin{bmatrix} 7.722 & 0 & -7.722 & 0 & 0 & 0 & 0 & 0 \\ 0 & 7.722 & 0 & 0 & 0 & -7.722 & 0 & 0 \\ 7.722 & 0 & 10.452 & 2.730 & -2.730 & -2.730 & 0 & 0 \\ 0 & 0 & 2.730 & 10.452 & -2.730 & -2.730 & 0 & -7.722 \\ 0 & 0 & -2.730 & -2.730 & 10.452 & 2.730 & -7.722 & 0 \\ 0 & -7.722 & -2.730 & -2.730 & 2.730 & 10.452 & 0 & 0 \\ 0 & 0 & 0 & 0 & -7.722 & 0 & 7.722 & 0 \\ 0 & 0 & 0 & -7.722 & 0 & 0 & 0 & 7.722 \end{bmatrix} \tag{23}$$

A estrutura foi modelada no software MASTAN2® e no código desenvolvido neste trabalho. Foram avaliados os deslocamentos nodais em x e y e a tensão em cada elemento da treliça. Os valores resultantes do algoritmo foram arredondados para conter as mesmas casas decimais oferecidas pelo software MASTAN2®, sendo então calculada a variação entre os dois modelos. O resultado da comparação está na Tabela 4.

Tabela 4 – Comparação de resultados da análise estrutural - MASTAN2

Variável	**Autor**	**MASTAN2**	**Variação (%)**
u_1	1.359 cm	1.359 cm	0%
v_1	—	—	—
u_2	1.125 cm	1.125 cm	0%
v_2	-0.4662 cm	-0.4662 cm	0%
u_3	—	—	—
v_3	—	—	—
u_4	—	—	—
v_4	—	—	—
σ_1	-80.357 MPa	-80.357 MPa	0%
σ_2	—	—	—
σ_3	113.642 MPa	113.642 MPa	0%
σ_4	-160.714 MPa	-160.714 MPa	0%
σ_5	—	—	—

Fonte: Autor

Pode-se observar que o modelo gerado possui uma assertividade de 100% quando comparado ao software MASTAN2®. Isso evidencia que o algoritmo está calibrado e pode ser utilizado para os propósitos deste estudo.

5 METODOLOGIA

Neste trabalho, três diferentes estruturas, que resultam em cinco casos particulares, serão otimizadas por meio do Search Group Algorithm. Os resultados obtidos pela otimização deverão então ser comparados com o resultado de outros autores que utilizaram diferentes algoritmos metaheurísticos para otimizar os mesmos problemas. Alguns indicadores estatísticos serão utilizados para efetuar tais comparações.

Ao longo das próximas seções, serão especificadas todas as características dos problemas, o processo de decisão das mesmas e o método utilizado para a análise.

5.1 DISCRETIZAÇÃO DO PROBLEMA DE OTIMIZAÇÃO EM TRELIÇAS

O problema de otimização utilizado neste trabalho visa encontrar uma treliça X com determinadas características (variáveis de projeto) de modo a minimizar seu peso (função objetivo), estando sujeita a diversas restrições. Assim como foi definido na seção 2, pode-se formular matematicamente o problema de otimização de treliças que foi desenvolvido ao longo deste estudo. Essa discretização é importante pois define de forma sistemática todas as características do problema de otimização, sendo essencial para a aplicação correta do método.

Encontrar,

$$\mathbf{x} = [A_1, A_2, ..., A_n, x_1, x_2, ..., x_m, y_1, y_2, ..., y_m]$$

Para minimizar,

$$f(\mathbf{x}) = \sum_{i=1}^{m} \rho_i A_i L_i \tag{24}$$

Sujeito a:

$g_1(\mathbf{x}) = A_j \in \Omega_{dim.}$ (Restrição dimensional);

$g_2(\mathbf{x}) = x_i, y_i \in \Omega_{geo.}$ (Restrição geométrica);

$g_3(\mathbf{x}) = |\sigma_j(\mathbf{x})| - \sigma_{adm} \leq 0$ (Tensão Admissível);

$g_4(\mathbf{x}) = |\delta_i(\mathbf{x})| - \delta_{max} \leq 0$ (Deslocamento);

$g_5(\mathbf{x}) = |\sigma_j^{flamb.}(\mathbf{x})| - \sigma_{adm} \leq 0$ (Flambagem);

$g_6 =$ Checagem de estabilidade da estrutura;

Onde x_m e y_2 são as coordenadas dos nós da treliça, ρ_i é a massa específica do material da estrutura, A_i é a área da seção transversal, L_i é o comprimento de cada barra, $\Omega_{dim.}$ e $\Omega_{geo.}$ são os conjuntos possíveis para as variáveis dimensional e geométrica, $\sigma_j(\mathbf{x})$ é a tensão na j-ésima barra da treliça, σ_{adm} é a tensão admissível do material, $\delta_i(\mathbf{x})$ é o deslocamento do i-ésimo nó da treliça, δ_{max} é o deslocamento máximo permitido e $\sigma_j^{flamb.}(\mathbf{x})$ é a tensão de flambagem para a i-ésima barra.

Como observado acima, cada indivíduo (ou estrutura) X pode ser formado por três tipos de variáveis em um problema de otimização de treliças. As variáveis $A_1, A_2, ..., A_n$ atuam na otimização **dimensional** e correspondem à área da seção transversal de um elemento da estrutura. Nos casos em que seja exigida simetria da estrutura, uma variável A_n pode representar uma dupla de elementos, fazendo com que estes possuam sempre os mesmos valores de seção transversal. Já as variáveis $x_1, x_2, ..., x_m$ e $y_1, y_2, ..., y_m$ representam as coordenadas x e y dos nós na otimização **geométrica**. Nesse caso, apenas são adicionados como variáveis de projetos aqueles nós que de fato serão movimentados na otimização. Por fim, uma terceira variável é definida ao atribuir valor 0 em uma variável dimensional A_n, representando a exclusão desta barra e sendo, portanto, uma variável **topológica**. O algoritmo deve reconhecer todas essas variáveis de forma a tratá-las corretamente ao longo da análise estrutural.

Após a seleção das variáveis de projeto e execução da análise estrutural, cada estrutura é então avaliada de acordo com seu peso, que é a função objetivo do problema. O somatório do peso de todas as barras resulta no peso global da estrutura (em kg). Os valores de comprimento da barra são calculados pelo algoritmo desenvolvido através das coordenadas dos nós da treliça, a área da seção transversal de cada barra é definida pela variável de projeto e a massa específica do material é um valor fixo para cada problema testado.

Ao total, seis restrições foram definidas para o problema de otimização de treliças, sendo utilizadas ou não, de acordo com o problema a ser resolvido. $g_1(\mathbf{x})$ e $g_2(\mathbf{x})$ restringem as variáveis dimensionais e geométricas por meio de limites superiores e inferiores específicos. Essa restrição foi tratada por meio do código de otimização, que gera indivíduos sempre dentro dos limites válidos definidos previamente para cada problema. $g_3(\mathbf{x})$ restringe a tensão aplicada em um elemento da treliça para que seja igual ou inferior ao valor da tensão admissível. O cálculo da tensão em cada elemento é feito pelo método de força e deslocamento e o resultado é tratado posteriormente por meio de penalidades. $g_4(\mathbf{x})$ restringe o deslocamento máximo dos nós devido à deformação da estrutura e também é calculado pelo método de forças e deslocamentos. Em geral, esse método, quando solicitado pelo problema, foi aplicado para a restrição do deslocamento vertical (eixo y). $g_5(\mathbf{x})$ restringe a flambagem de cada barra da treliça para que seja inferior à tensão admissível do material. O cálculo da flambagem, quando existente no problema, foi definido de acordo com os artigos da literatura que descrevem os problemas, para possibilitar a comparação

de resultados.

Por fim, g_6 restringe a falta de estabilidade da estrutura ao excluir barras e nós durante a otimização topológica. O tratamento dessa restrição ocorre de duas formas:

1. Ao excluir os nós que estão fixos (apoios) ou sob ação de forças
2. Quando o determinante da matriz de rigidez global for menor ou igual a zero

Caso o primeiro requisito não seja cumprido, a rotina de análise estrutural é interrompida e uma penalidade é aplicada. Caso contrário, a estrutura é analisada e o segundo requisito é testado logo após a formulação da matriz de rigidez. Caso não o atenda, a rotina é interrompida e uma penalidade é aplicada.

5.2 PENALIDADES

Nos problemas de otimização, uma série de restrições devem ser implementadas para que se obtenham resultados úteis. Nesse cenário, o transpasse de restrições como a tensão admissível e o deslocamento máximo representam uma estrutura não viável e, dessa forma, devem ser reconhecidas pelo algoritmo como estruturas ruins. Existem diferentes formas de alcançar isso, porém, neste estudo, optou-se por penalizar o transpasse de restrições com um acréscimo de peso na estrutura, tornando-a menos atraente.

Duas formas de penalização foram aplicadas:

- *Death Penalty* (pena de morte)
- *Penalização proporcional*

A função de penalidade mais simples é conhecida como *pena de morte* (MEZURA-MONTES; COELLO, 2011). O objetivo da penalização por *pena de morte* é eliminar da população de busca qualquer estrutura que transpasse a restrição. Sua aplicação se dá através da adição de um número grande (fator de penalização) ao peso da estrutura, como na equação abaixo:

$$p(\mathbf{x}) = p_{death} \cdot g_i \tag{25}$$

Onde p_{death} é o fator de penalidade por pena de morte (em kg) e g_i é o resultado do teste de restrição (1 para transpasse da restrição e 0 para restrição respeitada).

Um outro método também clássico para penalizações em otimização é a *penalização proporcional*, que aplica a penalização de forma proporcional ao transpasse da restrição. Por meio desse procedimento, o algoritmo pode favorecer estruturas que pouco ultrapassaram as restrições, em vez de eliminá-las por completo, garantindo uma maior diversidade nas populações iniciais e favorecendo a otimização em problemas mais complexos e com

muitos mínimos locais. O cálculo dessa penalização ocorre multiplicando a porcentagem do transpasse da restrição por um fator de penalização, como na equação abaixo:

$$p(\mathbf{x}) = p_{prop} \,.\, (1 + \%g_i(\mathbf{x}))^{ck} \tag{26}$$

Onde p_{prop} é o fator de penalidade proporcional (em kg) e $\%g_i(\mathbf{x})$ é a porcentagem de transpasse em relação à restrição. O valor ck é um coeficiente que altera o expoente do transpasse, porém neste estudo lhe será atribuído o valor 1.

Assim, a **penalização proporcional** pode ser utilizada para as restrições em que o cálculo da porcentagem de transpasse da restrição é possível. Neste trabalho, esse tipo de penalização foi utilizada para as restrições $g_3(\mathbf{x})$, $g_4(\mathbf{x})$ e $g_5(\mathbf{x})$ (tensão, deslocamento e flambagem). Já a penalidade por **pena de morte** foi utilizada para a restrição g_6 (checagem de estabilidade da estrutura).

A formulação da função objetivo pode ser adaptada da seguinte forma para incluir as penalidades aplicadas:

$$f(\mathbf{x}) = W(\mathbf{x}) + P(\mathbf{x}) \tag{27}$$

Onde W(x) é a função que calcula o peso da estrutura e P(x) a função que calcula as penalidades para as restrições ultrapassadas.

A função de penalidade pode ser definida como:

$$P(\mathbf{x}) = p_1(\mathbf{x}) + p_2(\mathbf{x}) \tag{28}$$

Onde $p_1(\mathbf{x})$ é a função da penalidade proporcional e $p_2(\mathbf{x})$ é a função da penalidade por pena de morte.

Por fim, as funções $p_1(\mathbf{x})$ e $p_2(\mathbf{x})$ são definidas como:

$$p_1(\mathbf{x}) = p_{prop}\left(1 + \sum_{i=1}^{m}\left[\left(\frac{\sigma_j(\mathbf{x}) - \sigma_{adm}}{\sigma_{adm}}\right) + \left(\frac{\delta i(\mathbf{x}) - \delta_{max}}{\delta_{max}}\right)\right]\right) \tag{29}$$

$$p_2(\mathbf{x}) = p_{death} \,.\, g_6 \tag{30}$$

5.3 MÉTODO DE ANÁLISE COMPARATIVA

Um dos importantes pontos a serem considerados ao se trabalhar com algoritmos metaheurísticos é a sua natureza estocástica, contendo aleatoriedade no momento de definir a população de busca inicial e na geração de todas as populações seguintes. Isso faz com que o resultado de um metaheurístico aplicado a um mesmo problema e com os mesmos parâmetros não seja igual em diferentes execuções. Assim sendo, a análise da performance de um algoritmo metaheurístico na resolução de um problema de estrutura não pode ser atrelada apenas ao ponto mínimo encontrado em uma execução individual.

Ao executar o algoritmo NIR vezes (número de execuções independentes), um campo amostral de resultados é criado e torna-se possível avaliar o desempenho dos metaheurísticos através de princípios da estatística. Dessa forma, não apenas o ponto ótimo entre todas as execuções é avaliado, mas também a média e a variação destes resultados, representando de forma mais fiel a performance do algoritmo.

Para garantir a validade estatística do estudo e seguindo o que foi implementado por Souza, Miguel e Lopez (2016) e outros autores comparados neste trabalho, cada problema será otimizado pelo algoritmo NIR = 100 vezes com os mesmos parâmetros. A partir dos resultados destas 100 amostras, alguns indicadores pré-determinados serão avaliados em comparação com a literatura, embasando a discussão sobre cada problema. Os indicadores utilizados foram:

- Peso Mínimo
- Peso Médio
- Coeficiente de Variação (C.O.V)
- Índice de Diversidade

O peso mínimo e médio referem-se, respectivamente, ao menor valor e à média dos pesos encontrados ao longo das 100 execuções independentes. O coeficiente de variação (C.O.V), também conhecido como desvio padrão relativo, é uma forma de analisar a dispersão em termos relativos (sem unidades vinculadas), pois utiliza a comparação do desvio padrão com a média dos valores. O coeficiente de variação é calculado por:

$$C.O.V(\%) = \frac{\sigma_x}{\overline{X}} \, . \, 100 \tag{31}$$

Onde σ_x é o desvio padrão e $\overline{X}$ é a média aritmética.

Já o Índice de Diversidade é uma métrica muito utilizada em algoritmos metaheurísticos, pois permite visualizar graficamente a diversidade da população a cada iteração. Nesse caso, observam-se claramente as fases de busca global (*exploration*) e local (*exploitation*), já que na segunda o índice de diversidade decresce consideravelmente para realizar buscas mais direcionadas. Para o cálculo do índice de diversidade foi utilizado o método proposto por Kaveh e Zolghadr (2014) e aplicado no SGA por Gonçalves, Lopez e Miguel (2015), definido como:

$$\text{Índice de Diversidade} = \frac{1}{n_{pop}} \sum_{i=1}^{n_{pop}} \sqrt{\sum_{j=1}^{n_{var}} \left(\frac{R_{1j} - P_{ij}}{x_j^{max} - x_j^{min}} \right)^2} \tag{32}$$

Onde n_{pop} é o tamanho da população, n_{var} é o número de variáveis, R_{1j} é a variável j da melhor estrutura da iteração, P_{ij} é o valor da variável j da i-ésima estrutura e x_j^{max} e x_j^{min} são, respectivamente, os limites superior e inferior da variável j.

Por fim, para analisar a geometria das estruturas resultantes deste estudo, um algoritmo na linguagem *Python* foi desenvolvido no presente estudo. O algoritmo realiza a impressão de uma representação gráfica da estrutura e compara sua geometria com a estrutura inicial do problema.

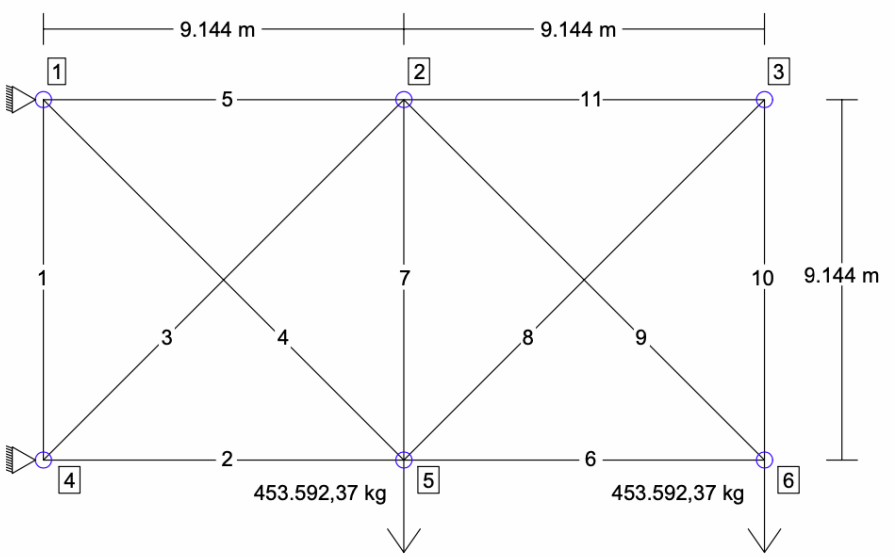
5.4 APRESENTAÇÃO DOS PROBLEMAS

Nesta seção serão detalhados os problemas de treliças planas a serem testados neste estudo. Como os problemas descritos abaixo são problemas de *benchmark* (para comparação), estes devem seguir as mesmas propriedades e restrições dos autores anteriores, viabilizando, assim, a sua correta comparação. Além disso, as unidades utilizadas por alguns autores foram convertidas para unidades do SI (Sistema Internacional de Unidades), por serem mais usuais no contexto brasileiro. Nota-se que a conversão dessas unidades não impacta o modelo de comparação utilizado. Por fim, para a escolha dos problemas, foram priorizadas treliças trabalhadas na literatura com uso de amostras estatísticas e indicadores como a média e C.O.V, o que garante uma melhor abordagem dos algoritmos metaheurísticos.

5.4.1 Problema 1 - Treliça de 11 barras - Dimensional, Geométrica e Topológica

O primeiro problema a ser estudado é uma treliça em balanço de 11 barras e 6 nós, com otimização dimensional, geométrica e topológica. Essa estrutura foi testada anteriormente com Algoritmos Genéticos (GA) por Rajan (1995) e por Balling, Briggs e Gillman (2006), com Harmony Search (HS) por Martini (2011), com Firefly Algorithm (FA) por Miguel, Lopez e Miguel (2013) e com Backtracking Search Algorithm (BSA) por Souza, Miguel e Lopez (2016). Ambos os apoios possuem restrição ao movimento no eixo x e y. São aplicadas duas cargas de 453.592,37 kg nos nós 5 e 6 da treliça. Na geometria inicial do problema, a distância entre nós consecutivos é de 914,4 cm.

Figura 20 – Problema 1 - Treliça em balanço de 11 barras



Fonte: Autor

Para as variáveis dimensionais são utilizados 32 valores discretos, contidos no conjunto $\Omega = \{10.45159, 11.61288, 15.35481, 16.90319, 18.58061, 19.93544, 20.19351, 21.80641, 23.41931, 24.77414, 24.96769, 26.96769, 28.96768, 30.96768, 32.06445, 33.03219, 37.03218, 46.58055, 51.41925, 74.1934, 87.0966, 89.67724, 91.61272, 99.9998, 103.2256, 121.29008, 128.38684, 141.9352, 147.74164, 170.9674, 193.5480, 216.1286\}$ cm^2. Já para as variáveis geométricas, os nós 1, 2 e 3 podem se movimentar de forma independente no eixo y, assumindo valores entre 4,572m e 25,40m (a coordenada [0,0] foi posicionada no nó 4). Os três nós inferiores não possuem variáveis geométricas e permanecem fixos ao longo de toda a otimização. Assim, existe um total de 14 variáveis de projeto, sendo 11 variáveis dimensionais e 3 variáveis geométricas.

Vale também comentar que cada uma das 11 barras pode assumir valores de seção transversal de forma independente, não havendo simetria para a otimização dimensional. A otimização topológica também foi utilizada nessa treliça, sendo que a exclusão de cada barra é conseguida ao atribuir-se o valor 0 à variável dimensional.

Além de limites superiores e inferiores das variáveis dimensionais, foram também aplicadas ao problema 1 restrições quanto ao deslocamento de nós no eixo y e à tensão admissível do material. Os valores assumidos para estas restrições, bem como os valores para as propriedades estruturais do problema estão declarados na tabela 5

Tabela 5 – Valores das propriedades estruturais inseridas no problema 1

Propriedade	**Valor utilizado**
E	68.947591 GPa
ρ	2767.990 kg/m^3
σ_{adm}	172.369 MPa
y_{disp}	50.8 mm

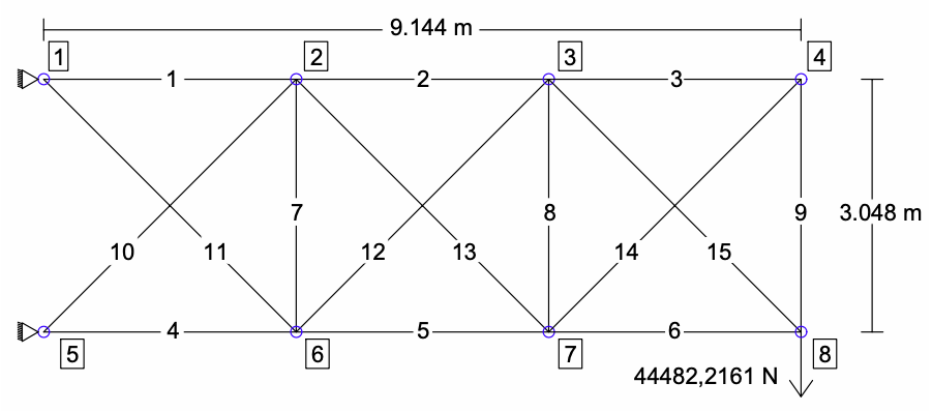
Fonte: Autor

Onde E é o módulo de elasticidade, ρ é a massa específica do material, σ_{adm} é a tensão admissível e y_{disp} é o deslocamento admissível no eixo y.

5.4.2 Problema 2a - Treliça de 15 barras - Dimensional e Geométrica

O problema 2 trata-se de uma treliça em balanço com 15 barras e 8 nós. Essa estrutura foi estudada anteriormente por diversos autores. Serão comparados aqui o uso de Algoritmos Genéticos por Wu e Chow (1995), Firefly Algorithm (FA) por Miguel, Lopez e Miguel (2013) e Backtracking Search Algorithm (BSA) por Souza, Miguel e Lopez (2016). A distância entre nós consecutivos é de 304,8 cm e os dois apoios à esquerda da estrutura possuem restrição de movimento nas coordenadas x e y. O peso P aplicado ao nó 8 tem intensidade de 44482,2161 N e sentido negativo em relação a y. O problema "a" dessa estrutura utiliza apenas a otimização dimensional e geométrica.

Figura 21 – Problema 2 - Treliça em balanço de 15 barras



Fonte: Autor

No problema analisado, a otimização dimensional é discreta e utiliza os 32 valores de seção transversal do conjunto $\Omega = \{0.716, 0.910, 1.123, 1.419, 1.742, 1.852, 2.239, 2.839, 3.478, 6.155, 6.974, 7.574, 8.600, 9.600, 11.381, 13.819, 17.400,18.064, 20.200, 23.000, 24.600, 31.000, 38.400, 42.400, 46.400, 55.000, 60.000, 70.000, 86.000, 92.193, 110.774, 123.742\}$ cm^2. Já para as variáveis geométricas, os nós 2 e 6 e os nós 3 e 7 se movimentam de forma

conjunta no eixo x, assumindo valores entre 2,54m e 3,556m para os nós 2 e 6, e valores entre 5,588m e 6,604m para os nós 3 e 7. No eixo y cada nó se movimenta de forma livre, com exceção dos nós 1 e 5, que permanecem fixos durante toda a otimização. A tabela 6 descreve os limites superiores e inferiores de todas as variáveis geométricas inseridas. Ao total, o problema de otimização contém 23 variáveis, sendo 15 variáveis dimensionais e 8 variáveis geométricas.

Tabela 6 – Limites superior e inferior para variáveis geométricas no problema 2

Nó	**Eixo**	**Limite Superior (m)**	**Limite inferior (m)**
2 e 6	x	3.556	2.54
3 e 7	x	6.604	5.588
2	y	3.556	2.54
3	y	5.588	2.54
4	y	2.286	1.27
6	y	0.508	-0.508
7	y	0.508	-0.508
8	y	1.524	0.508

Fonte: Autor

Neste problema, foram utilizadas restrições de tensão admissível, deslocamento de nós no eixo y e restrição devido à flambagem. O cálculo da tensão de flambagem foi introduzido ao algoritmo com base na equação simplificada:

$$\sigma_{cr} = \frac{100EA_i}{8L_i^2} \leq \sigma_{adm} \tag{33}$$

Onde E é o módulo de elasticidade do material, A_i e L_i são respectivamente a área e o comprimento da barra i e σ_{adm} é a tensão admissível do problema. As propriedades utilizadas nesse problema estão descritas na tabela abaixo:

Tabela 7 – Valores das propriedades estruturais inseridas no problema 2

Propriedade	**Valor utilizado**
E	68.947591 GPa
ρ	2767.990 kg/m^3
σ_{adm}	172.369 MPa

Fonte: Autor

5.4.3 Problema 2b - Treliça de 15 barras - Dimensional, Geométrica e Topológica

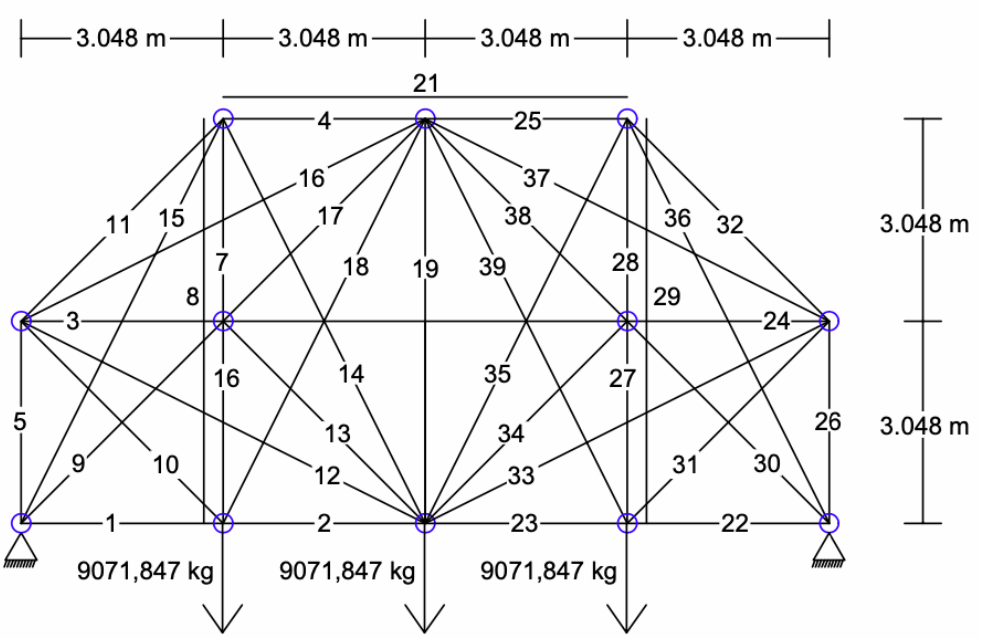
O caso b do problema 2 foi proposto por Miguel, Lopez e Miguel (2013) e aplicado também por Souza, Miguel e Lopez (2016), inserindo a otimização topológica na estrutura. Neste caso, todas as barras podem ser removidas de forma independente ao alterar o valor da variável dimensional para zero. É importante ressaltar que a estabilidade da estrutura

foi testada como uma restrição do problema, aplicando a penalização por pena de morte nos casos com transpasse dessa restrição.

5.4.4 Problema 3a - Treliça de 39 barras - Dimensional e Topológica

O problema 3 trata-se de uma treliça biapoiada de 39 barras e 12 nós. Esta estrutura foi estudada anteriormente por Deb e Gulati (2000), Luh e Lin (2008), Wu e Tseng (2010), Luh e Lin (2011), Miguel, Lopez e Miguel (2013), Ahrari, Atai e Deb (2014) e Souza, Miguel e Lopez (2016). Os nós da estrutura estão organizados em três alinhamentos horizontais e a distância entre nós consecutivos é de 304,8 cm. O apoio esquerdo possui restrição nos dois eixos, enquanto o apoio direito possui restrição apenas no eixo y. São aplicadas três cargas atuantes nos nós 2, 3 e 4, todas com massa de 9071,847 kg. O primeiro caso deste problema envolve a otimização dimensional e topológica. Para o problema será utilizado, como outros autores, OFE = 50000 (avaliações da função objetivo).

Figura 22 – Problema 3 - Treliça biapoiada de 39 barras



Fonte: Autor

Para as variáveis dimensionais são utilizados valores contínuos limitados entre -1,4516 x 10^{-4} e 1,4516 x 10^{-3}. Por ser utilizada a otimização dimensional com valores contínuos, é definido também um valor limite para a exclusão de barras na otimização topológica de 3,2258 x 10^{-5}. Abaixo deste valor, a barra é excluída.

Além disso, a otimização deve ocorrer de forma a manter a simetria da estrutura. Consequentemente, cada variável dimensional pode representar mais de uma barra, reali-

zando a aplicação da mesma seção transversal para barras espelhadas. Cada uma das 21 variáveis dimensionais estão definidas na tabela 8.

Tabela 8 – Variáveis dimensionais com simetria para o problema 3

Nº Variável	**Barras correspondentes**	**Nº Variável**	**Barras correspondentes**
1	1 e 22	12	12 e 33
2	2 e 23	13	13 e 34
3	3 e 24	14	14 e 35
4	4 e 25	15	15 e 36
5	5 e 26	16	16 e 37
6	6 e 27	17	17 e 38
7	7 e 28	18	18 e 39
8	8 e 29	19	19
9	9 e 30	20	20
10	10 e 31	21	21
11	11 e 32		

Fonte: Autor

As restrições utilizadas neste exemplo, além de limites superiores e inferiores das variáveis dimensionais, foram o deslocamento de nós no eixo y e tensão admissível do material. Os valores assumidos para estas restrições, bem como os valores para as propriedades estruturais do problema estão declarados na tabela abaixo:

Tabela 9 – Valores das propriedades estruturais inseridas no problema 3

Propriedade	**Valor utilizado**
E	68.947591 GPa
ρ	2767.990 kg/m^3
σ_{adm}	137.895 MPa
y_{disp}	50.8 mm

Fonte: Autor

5.4.5 Problema 3b - Treliça de 39 barras - Dimensional, Geométrica e Topológica

Já para o caso b do problema 3, é adicionada a otimização geométrica de forma simultânea às outras otimizações. Todos os autores citados anteriormente para o problema 3a também aplicaram o caso atual. Para a otimização geométrica, os nós 1, 2, 3, 4 e 5 permanecem fixos, não sendo alocadas como variáveis do problema. O restante dos nós pode variar tanto no eixo y, quanto no eixo x, a uma distância de (-304.8, 304.8) cm da sua posição original.

Com o objetivo de manter a simetria na estrutura, as variáveis geométricas precisam ser alocadas de forma a corresponder com nós espelhados. A tabela 10 destaca cada uma das 7 variáveis geométricas e sua variação superior e inferior.

Tabela 10 – Limites superior e inferior para variáveis geométricas no problema 3

Nó	**Eixo**	**Variação Superior (m)**	**Variação inferior (m)**
6 e 9	x	3.047	-3.047
7 e 8	x	3.047	-3.047
10 e 12	x	3.047	-3.047
6 e 9	y	6.095	0
7 e 8	y	6.095	0
10 e 12	y	9.143	3.049
11	y	9.143	3.049

Fonte: Autor

6 RESULTADOS

6.1 PROBLEMA 1

Para a otimização do primeiro problema, foi utilizado um critério de parada de 50.000 OFEs (avaliações da função objetivo). Assim, um conjunto de parâmetros foram escolhidos para melhor adequar o algoritmo ao problema proposto, com o intuito de obter um processo de otimização eficiente. O conjunto de parâmetros está detalhado na tabela 11. É importante ressaltar que, para todas as estruturas avaliadas neste estudo, foi atribuído um valor de 0.2 ao parâmetro SG_{ratio} (taxa de grupo de busca).

Tabela 11 – Parâmetros utilizados para o problema 1

Parâmetro	**Valor**	**Parâmetro**	**Valor**
α_{min}	0.075	It_{ratio}^{global}	0.6
α_0	1	$N_{perturbed}$	4
$N_{iterations}$	700	$penal_{death}$	10^5
n_{pop}	85	$penal_{prop}$	300

Fonte: Autor

Os resultados deste problema foram comparados com outros cinco estudos, conforme exposto na tabela 21. Os autores Rajan (1995), Balling, Briggs e Gillman (2006) e Martini (2011) não demonstraram em seus resultados o peso médio e o coeficiente de variação obtidos em seus estudos, assim, a comparação para esses casos se dará apenas em relação ao peso mínimo.

Tabela 12 – Comparação dos resultados obtidos para o problema 1

Resultado	**Rajan (1995)**	**Balling, Briggs e Gillman (2006)**	**Martini (2011)**	**Miguel, Lopez e Miguel (2013)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
OFE	3840	500000	4075	50000	50000	50000
Peso mínimo (kg)	1475.999	1241.029	1315.418	1227.04	1227.07	1227.04
Peso médio (kg)	-	-	-	1268.285	1265.654	1273.394
C.O.V (%)	-	-	-	2.12	2.02	1.78

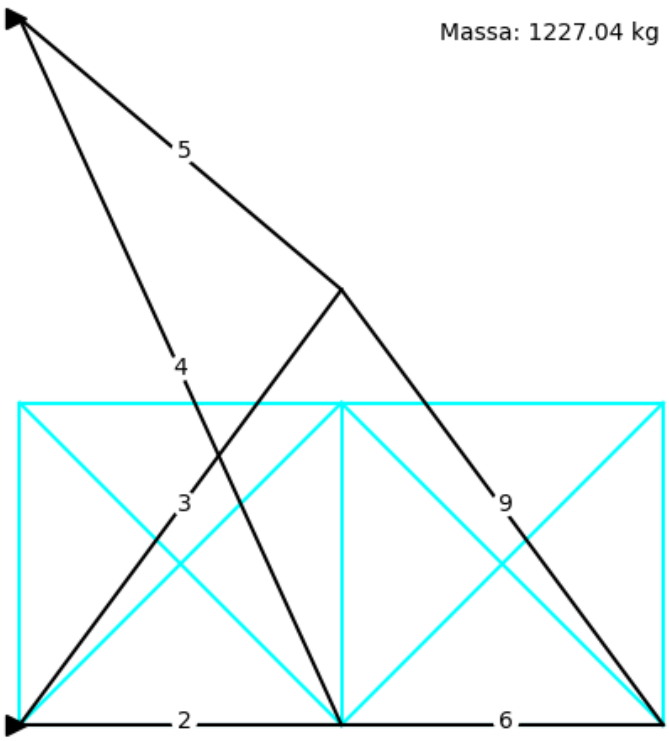
Fonte: Autor

Ao longo das 50.000 avaliações da função objetivo e das 100 execuções independentes do algoritmo, o SGA desenvolvido neste estudo encontrou uma estrutura com

peso mínimo de 1227,04 kg, o menor peso encontrado entre os autores comparados. Essa mesma estrutura foi também encontrada por Miguel, Lopez e Miguel (2013), enquanto que Souza, Miguel e Lopez (2016) atingiu um valor bastante próximo (1227,07 kg). Isso pode ser um indicativo de que este problema envolve uma estrutura menos robusta ou com um mínimo bem definido. É importante ressaltar que Rajan (1995) e Martini (2011) utilizaram um número consideravelmente menor de avaliações da função objetivo (abaixo de 4500), resultando em mínimos menos competitivos.

O peso médio obtido ao longo das 100 execuções foi o maior quando comparado com Miguel, Lopez e Miguel (2013) e Souza, Miguel e Lopez (2016). Apesar disso, pode-se observar que o coeficiente de variação obtido (1.78%) é menor que ambos os autores supracitados, o que garante que o algoritmo obteve resultados mais próximos entre si. A figura 23 é uma representação gráfica da otimização realizada. Em azul, é representada a estrutura inicial do problema e, em preto, a estrutura resultante.

Figura 23 – Melhor estrutura encontrada para o problema 1

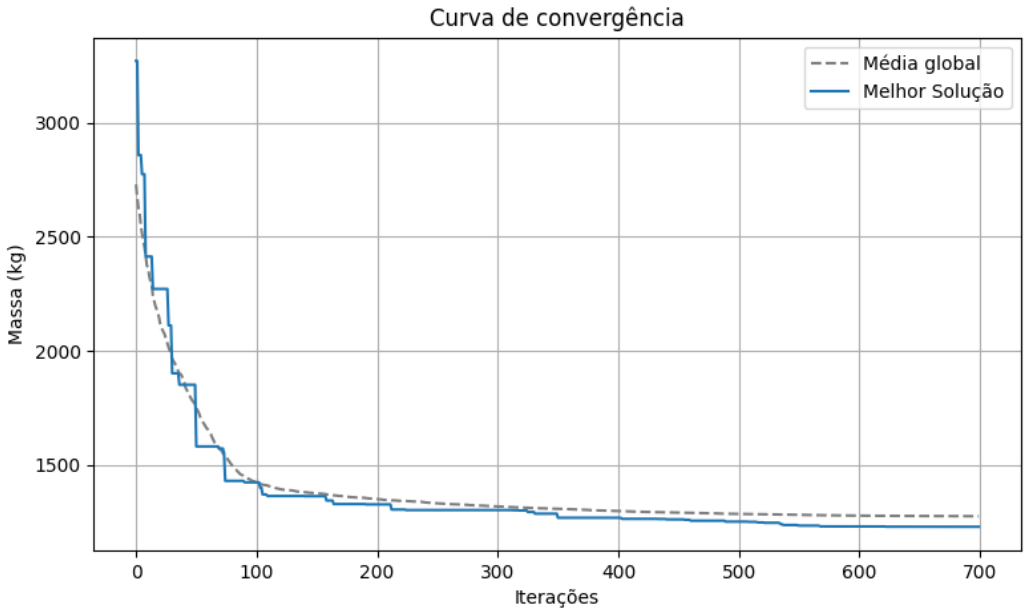


Fonte: Autor

Por fim, são apresentados abaixo a curva de convergência (Figura 24) e o índice de diversidade para cada iteração (Figura 25). A curva de convergência é gerada pelo peso mínimo obtido em cada uma das 700 iterações do algoritmo. Aqui são apresentadas a curva da melhor solução (em azul) e a curva com os valores médios de peso mínimo de

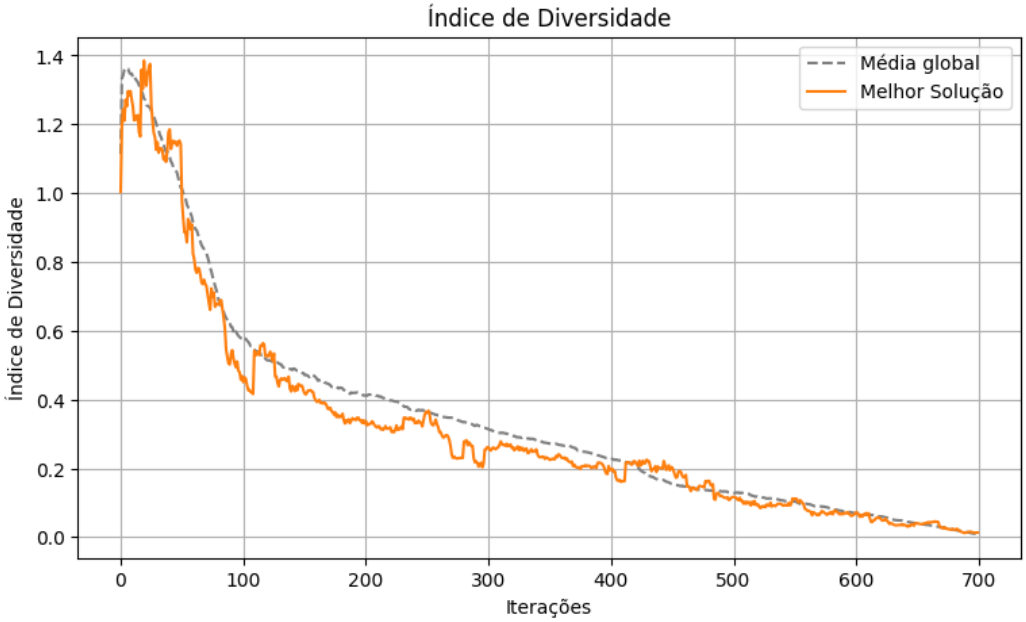
cada iteração (linha pontilhada). Já o índice de diversidade foi calculado conforme descrito no capítulo 5 e representa a diversidade da população de cada iteração.

Figura 24 – Curva de convergência para o problema 1



Fonte: Autor

Figura 25 – Índice de Diversidade para o problema 1



Fonte: Autor

6.2 PROBLEMA 2A

Para o problema de otimização 2a, foi utilizado um critério de parada de 8.000 OFEs (avaliações da função objetivo). Esse número, quando comparado ao problema anterior, é consideravelmente menor e, por isso, exigiu uma maior atenção para a escolha dos parâmetros. Os parâmetros utilizados no problema são apresentados na tabela abaixo:

Tabela 13 – Parâmetros utilizados para o problema 2a

Parâmetro	**Valor**	**Parâmetro**	**Valor**
α_{min}	0.075	It_{ratio}^{global}	0.6
α_0	1	$N_{perturbed}$	4
$N_{iterations}$	130	$penal_{death}$	10^6
n_{pop}	76	$penal_{prop}$	20

Fonte: Autor

Este problema foi comparado com outros três estudos que utilizaram Algoritmos Genéticos (GA), Firefly (FA) e Backtracking Search (BSA). A tabela 22 detalha os resultados obtidos por todos esses autores.

Tabela 14 – Comparação dos resultados obtidos para o problema 2a

Resultado	**Wu e Chow (1995)**	**Miguel, Lopez e Miguel (2013)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
OFE	-	8000	8000	8000
Peso mínimo (kg)	182.506	62.627	59.641	64.032
Peso médio (kg)	-	69.948	64.049	69.856
C.O.V (%)	-	3.85	3.96	2.87

Fonte: Autor

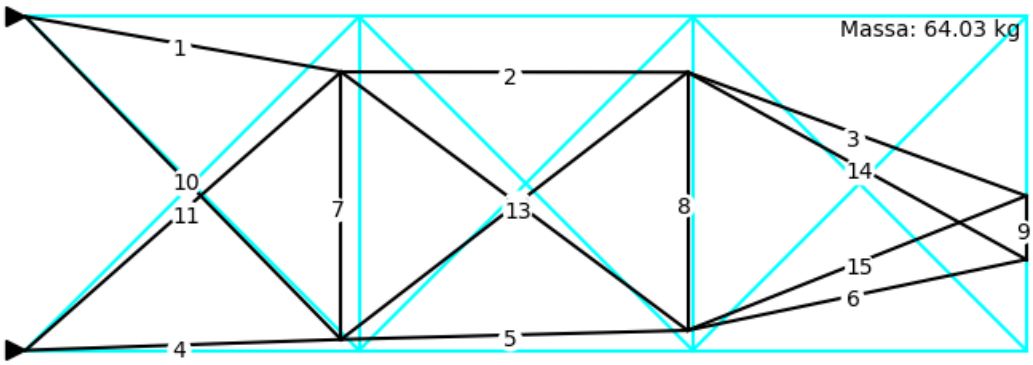
No estudo realizado por Wu e Chow (1995), não há informações relacionadas ao número de execuções do algoritmo, bem como a média e variação destes resultados. Ainda, este autor obteve um peso consideravelmente maior que os demais estudos, dificultando as comparações. O presente estudo obteve um peso mínimo de 64,03 kg, um valor 2,19% maior que Miguel, Lopez e Miguel (2013) e 6,86% maior que Souza, Miguel e Lopez (2016). Dessa forma, é possível dizer que o algoritmo SGA elaborado neste estudo obteve problemas em alcançar mínimos globais para essa estrutura. Uma possível justificativa para isso está relacionada à dificuldade na escolha de um fator de penalização $penal_{prop}$ aplicável para

este problema. Com a aplicação de valores muito baixos para o parâmetro, a otimização passava a favorecer estruturas inválidas de acordo com as restrições. Já a aplicação de valores altos impedia a exploração do domínio em regiões próximas às restrições.

Por outro lado, foi obtido o valor de 69,856 kg para o peso médio entre as avaliações independentes, isto é, foi obtido um peso médio menor que o encontrado por Miguel, Lopez e Miguel (2013). Além disso, foi encontrado um coeficiente de variação de 2,87%, menor que o indicado pelos outros autores. Sendo assim, essas informações são um bom indicativo de que o algoritmo desenvolvido possui resultados comparáveis ao estudo supracitado.

Na figura 26, o melhor resultado encontrado é representado graficamente em comparação com a estrutura inicial. Percebe-se que os nós mais distantes dos apoios, onde a carga está sendo aplicada, tendem a se agrupar no eixo horizontal da estrutura, resultando em uma melhor distribuição dos esforços.

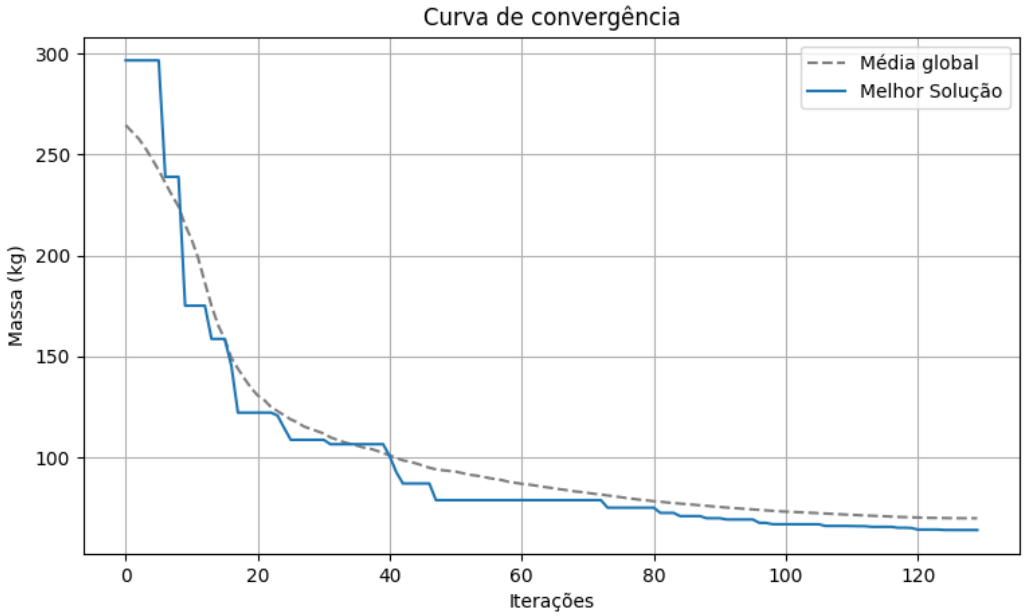
Figura 26 – Melhor estrutura encontrada para o problema 2a



Fonte: Autor

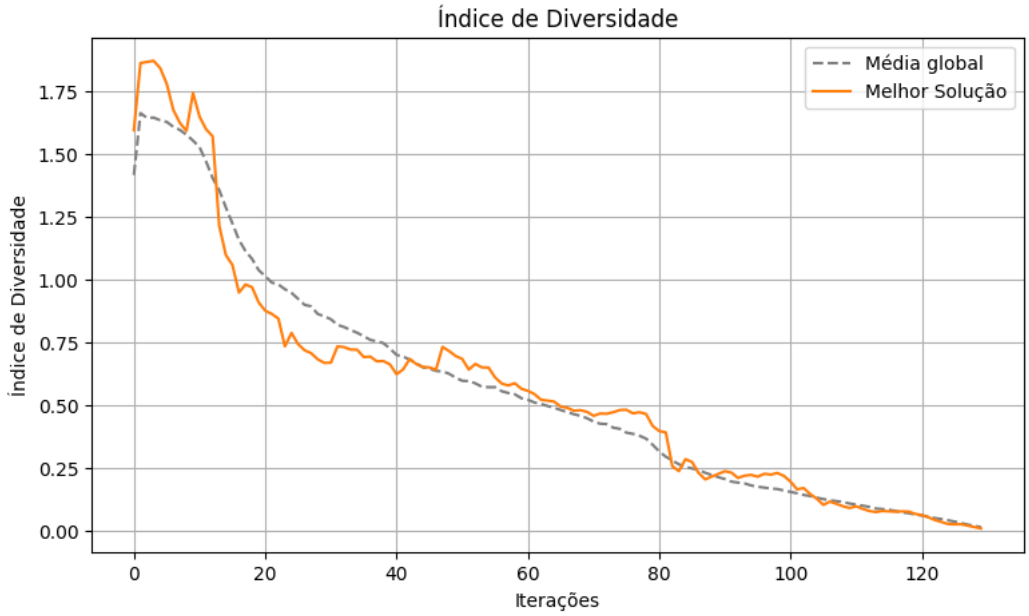
Nas figuras 27 e 28, são apresentadas a curva de convergência e o índice de diversidade da melhor estrutura e da média global do problema.

Figura 27 – Curva de convergência para o problema 2a



Fonte: Autor

Figura 28 – Índice de Diversidade para o problema 2a



Fonte: Autor

6.3 PROBLEMA 2B

Assim como no caso "a" do problema 2, foi utilizado um critério de parada de 8.000 OFEs (avaliações da função objetivo) na resolução deste problema. Os parâmetros utilizados no problema são apresentados na tabela abaixo:

Tabela 15 – Parâmetros utilizados para o problema 2b

Parâmetro	**Valor**	**Parâmetro**	**Valor**
α_{min}	0.075	It_{ratio}^{global}	0.6
α_0	1	$N_{perturbed}$	4
$N_{iterations}$	130	$penal_{death}$	10^5
n_{pop}	76	$penal_{prop}$	15

Fonte: Autor

O problema 2b foi comparado com outros dois estudos que utilizaram Firefly (FA) e Backtracking Search (BSA). A tabela 23 detalha os resultados obtidos por seus autores.

Tabela 16 – Comparação dos resultados obtidos para o problema 2b

Resultado	**Miguel, Lopez e Miguel (2013)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
OFE	8000	8000	8000
Peso mínimo (kg)	56.803	54.827	56.016
Peso médio (kg)	69.223	60.187	66.973
C.O.V (%)	8.69	7.88	6.01

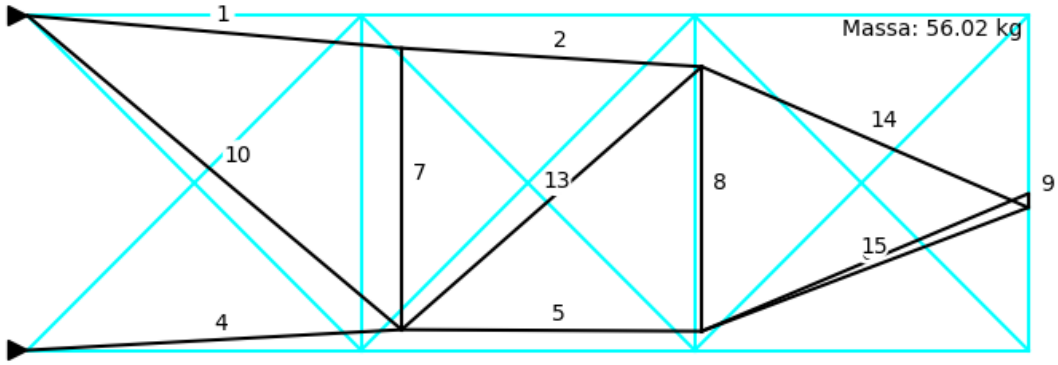
Fonte: Autor

Foi obtido um peso mínimo de 56,016 kg neste estudo, um valor que fica entre os dois autores comparados. Com relação a Miguel, Lopez e Miguel (2013), o peso mínimo foi reduzido em 1.41% neste estudo. Apesar disso, o mínimo encontrado ainda é 2.12% maior que o encontrado por Souza, Miguel e Lopez (2016). O mesmo ocorre para o peso médio, em que o valor resultante é 3,36% menor que o valor de Miguel, Lopez e Miguel (2013), porém 10,13% maior que o valor de Souza, Miguel e Lopez (2016). Novamente, em se tratando do coeficiente de variação obtido, o valor é consideravelmente menor que o obtido pelos outros autores.

De forma geral, houve uma boa convergência do método de otimização, resultando em valores mínimos e médios competitivos. Na Figura 29 a melhor estrutura encontrada é

apresentada graficamente. Quando comparada à estrutura gerada pelo caso "a" é perceptível que os nós à direita encontram-se mais unidos no eixo horizontal da estrutura. Além disso, foi excluído um total de três barras no processo de otimização topológica (elementos 3, 11 e 12), reduzindo consideravelmente o peso da estrutura.

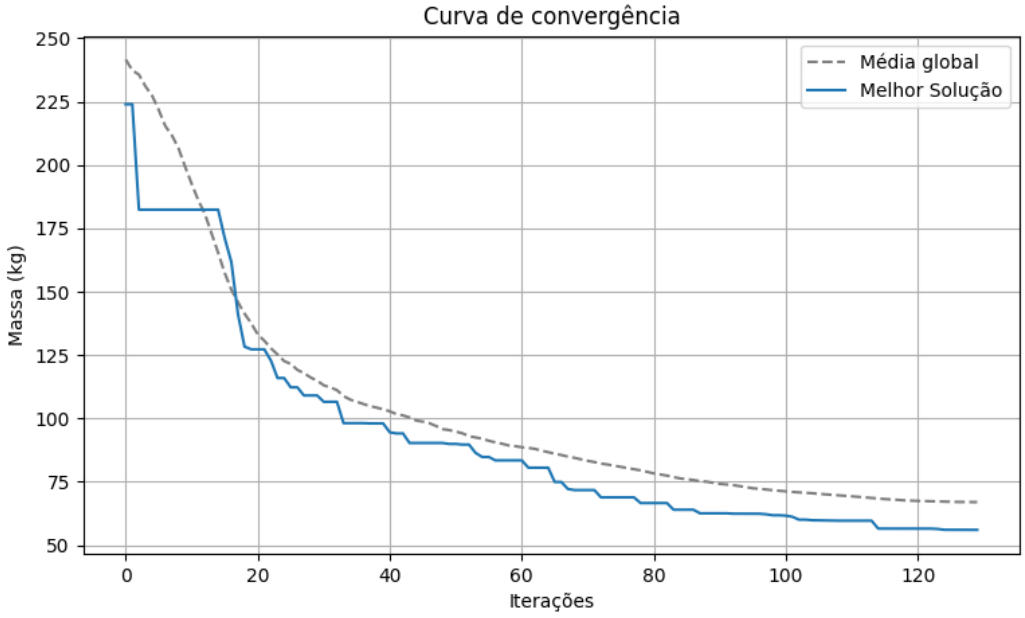
Figura 29 – Melhor estrutura encontrada para o problema 2b



Fonte: Autor

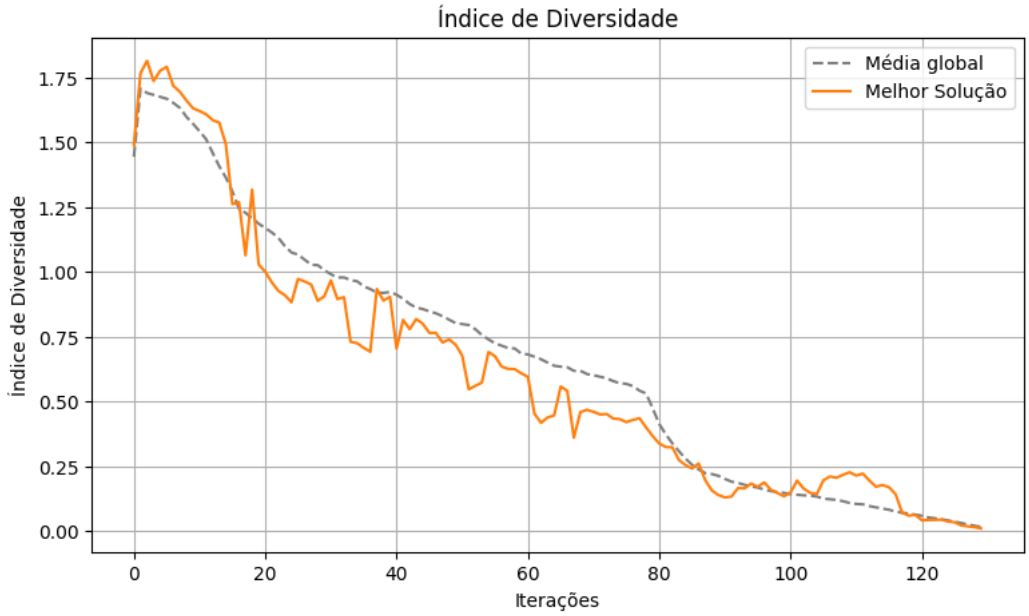
Por fim, as figuras 30 e 31 demonstram o comportamento do processo de otimização com relação ao peso mínimo (convergência) e diversidade da população (índice de diversidade). Nota-se que a curva de convergência da melhor solução apresenta um peso consideravelmente menor ao longo de grande parte do processo de otimização.

Figura 30 – Curva de convergência para o problema 2b



Fonte: Autor

Figura 31 – Índice de Diversidade para o problema 2b



Fonte: Autor

6.4 PROBLEMA 3A

O problema 3a corresponde a uma treliça biapoiada de 39 barras com otimização dimensional e topológica. Sendo assim, não é permitida, para este caso, a movimentação dos nós. Para a resolução do problema, foi utilizado um critério de parada de 50.000 OFEs (avaliações da função objetivo). Os parâmetros utilizados no problema são apresentados na tabela abaixo:

Tabela 17 – Parâmetros utilizados para o problema 3a

Parâmetro	**Valor**	**Parâmetro**	**Valor**
α_{min}	0.075	It_{ratio}^{global}	0.6
α_0	1	$N_{perturbed}$	4
$N_{iterations}$	700	$penal_{death}$	10^5
n_{pop}	85	$penal_{prop}$	20

Fonte: Autor

O problema 3a foi estudado anteriormente nos artigos de Deb e Gulati (2000), Luh e Lin (2008), Wu e Tseng (2010), Luh e Lin (2011), Miguel, Lopez e Miguel (2013) e Souza, Miguel e Lopez (2016), os quais serão utilizados para comparação com os resultados obtidos neste estudo. A tabela 24 apresenta todos os resultados obtidos por esses autores.

Tabela 18 – Comparação dos resultados obtidos para o problema 3a

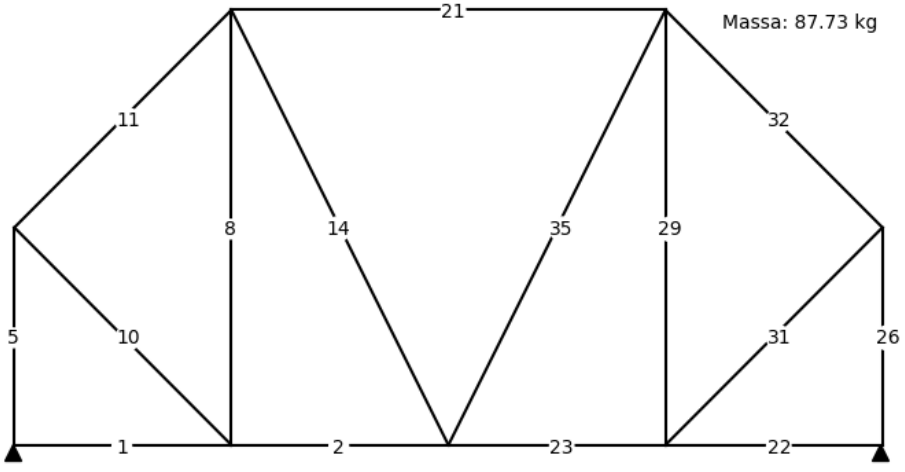
Resultado	**Deb e Gulati (2000)**	**Luh e Lin (2008)**	**Wu e Tseng (2010)**	**Luh e Lin (2011)**	**Miguel, Lopez e Miguel (2013)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
OFE	504000	303600	32300	262500	50000	50000	50000
Peso mínimo (kg)	89.152	87.758	87.634	87.549	87.792	87.637	87.728
Peso médio (kg)	-	-	-	-	100.552	93.284	89.199
C.O.V (%)	-	-	-	-	12.9	5.28	1.82

Fonte: Autor

Conforme observado nos resultados acima, este estudo obteve um valor de peso mínimo de 87,727 kg, atingindo melhores mínimos que alguns outros autores, a saber: Deb e Gulati (2000), Luh e Lin (2008) e Miguel, Lopez e Miguel (2013). O peso mínimo obtido foi ligeiramente maior que o obtido pelos autores Wu e Tseng (2010), Luh e Lin (2011) e Souza, Miguel e Lopez (2016). Apesar disso, quando comparamos os valores de média e coeficiente de variação, o presente estudo atingiu uma consistência melhor, atingindo de forma mais precisa bons valores de mínimo. É o que se comprova ao observar o valor médio de 89,199 kg, 4,58% menor que o encontrado por Souza, Miguel e Lopez (2016), e o coeficiente de variação de 1,82%, menor que o obtido pelos outros autores.

A Figura 32 representa graficamente a melhor estrutura encontrada, onde se pode observar que apenas 15 barras são mantidas (exclusão de 24 barras).

Figura 32 – Melhor estrutura encontrada para o problema 3a

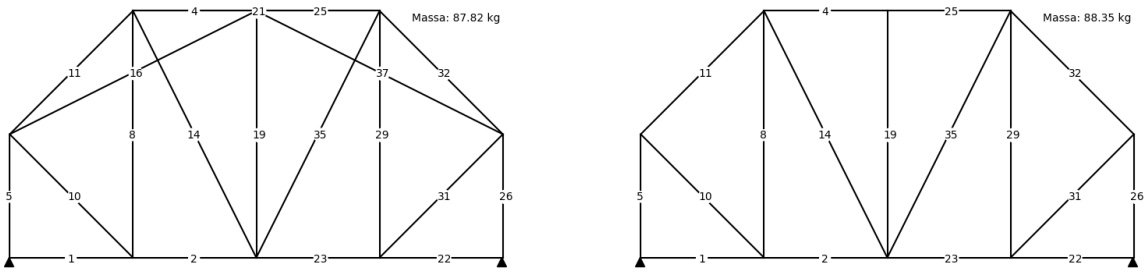


Fonte: Autor

Devido à grande quantidade de barras que podem ser excluídas neste problema, uma variedade de topologias pode ser observada entre os resultados obtidos. Nas figuras 33a e 33b são apresentadas duas alternativas de *design* obtidas ao longo do processo de otimização.

Figura 33 – Estruturas alternativas encontradas para o problema 3a

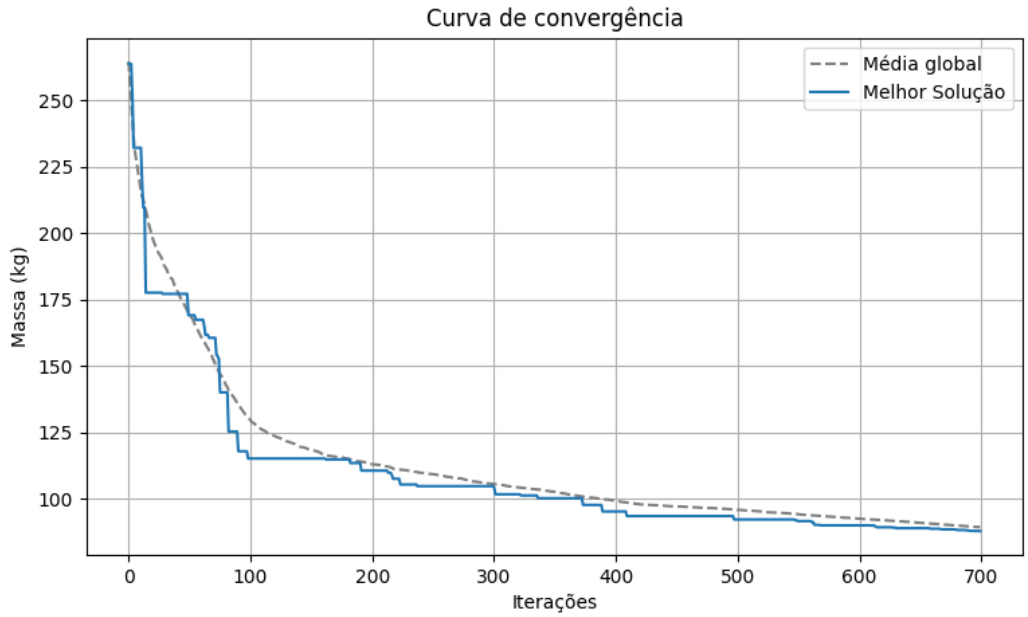
(a) 87,82 kg (b) 88,35 kg



Fonte: Autor

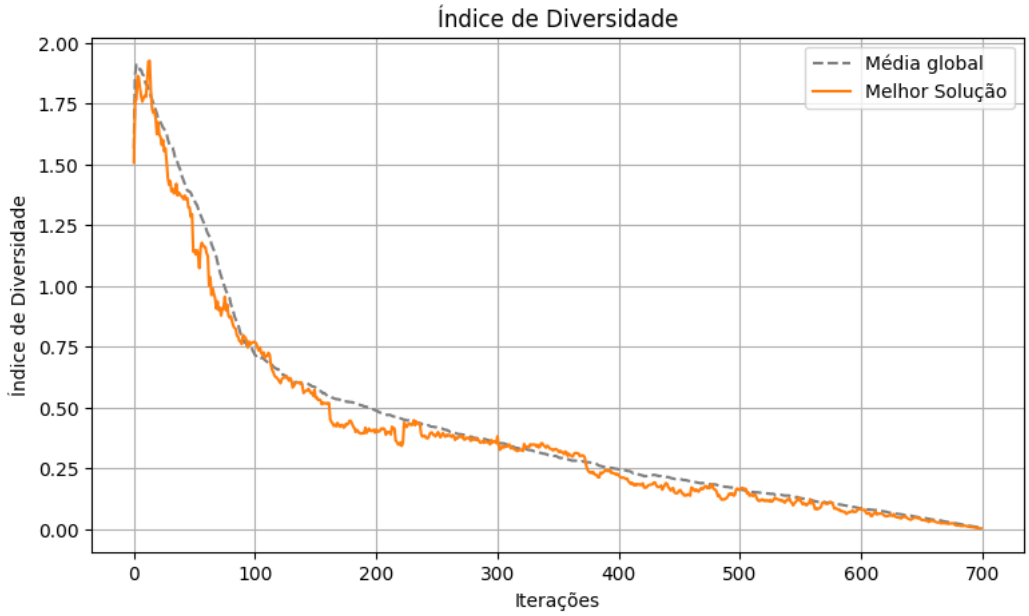
Por fim, são apresentadas, nas Figuras 34 e 35, a curva de convergência e índice de diversidade para cada uma das 700 iterações definidas e aplicadas neste problema. Nota-se que a melhor solução e a média global em ambos os gráficos estão bastante próximos, mais uma evidência da consistência do algoritmo ao longo do processo de otimização.

Figura 34 – Curva de convergência para o problema 3a



Fonte: Autor

Figura 35 – Índice de Diversidade para o problema 3a



Fonte: Autor

6.5 PROBLEMA 3B

O problema 3b utiliza 50.000 OFEs e os mesmos parâmetros utilizados no caso anterior. Esses parâmetros estão novamente detalhados na tabela 19. A diferença deste caso é a inclusão de variáveis geométricas, permitindo a movimentação dos nós e tornando o problema mais versátil.

Tabela 19 – Parâmetros utilizados para o problema 3b

Parâmetro	**Valor**	**Parâmetro**	**Valor**
α_{min}	0.075	It_{ratio}^{global}	0.6
α_0	1	$N_{perturbed}$	4
$N_{iterations}$	700	$penal_{death}$	10^5
n_{pop}	85	$penal_{prop}$	20

Fonte: Autor

Os mesmos autores do problema anterior foram comparados para este caso, porém com a inclusão de um novo estudo realizado por Ahrari, Atai e Deb (2014). A tabela 25 detalha os resultados obtidos por esses autores.

Tabela 20 – Comparação dos resultados obtidos para o problema 3b

Resultado	**Deb e Gulati (2000)**	**Luh e Lin (2008)**	**Wu e Tseng (2010)**	**Luh e Lin (2011)**	**Miguel, Lopez e Miguel (2013)**	**Ahrari, Atai e Deb (2014)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
OFE	504000	453600	137200	262500	50000	40256	50000	50000
Peso mínimo (kg)	87.176	85.607	85.469	85.541	86.774	82.091	86.753	86.356
Peso médio (kg)	-	-	-	-	94.478	-	100.59	89.095
C.O.V (%)	-	-	-	-	5.3	-	4.6	2.8

Fonte: Autor

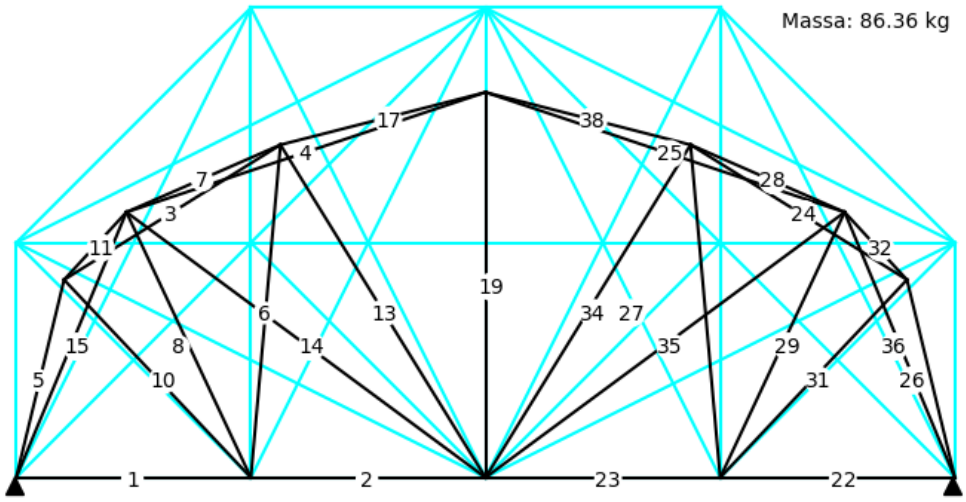
O SGA implementado para este problema obteve um peso mínimo de 86,356 kg, um peso médio de 89,095 kg e um coeficiente de variação de 2,8% ao longo de 100 avaliações independentes do algoritmo. O algoritmo, apesar de ter atingido bons resultados, teve uma colocação pior em relação a Ahrari, Atai e Deb (2014), que obtiveram um valor de 82,091 kg ao longo de 40.256 OFEs, e Luh e Lin (2008) e Wu e Tseng (2010), que obtiveram valores mínimos em torno de 85,5 kg. Vale notar que estes últimos utilizaram um número de avaliações da função objetiva mais de duas vezes maior que o do presente estudo, dificultando a comparação direta dos resultados obtidos. Outro fator que dificulta a análise comparativa com esses autores é a falta dos indicadores de média e coeficiente de variação, sem os quais é difícil afirmar a consistência dos algoritmos apresentados em obter resultados com baixo peso mínimo.

Fazendo a comparação com os estudos realizados por Miguel, Lopez e Miguel (2013) e Souza, Miguel e Lopez (2016), o SGA aplicado neste estudo atingiu os menores valores em peso mínimo, peso médio e coeficiente de variação, o que indica que o algoritmo conseguiu atingir mínimos competitivos e ainda consistentes a cada execução independente. Assim, conclui-se que o SGA alcançou resultados bastante satisfatórios para este problema.

A estrutura mínima encontrada para o problema 3b está detalhada na Figura 36. Na estrutura, observa-se que um total de 10 barras foram excluídas na otimização topológica. Além disso, a Figura 37 apresenta outras três geometrias alternativas encontradas ao longo da otimização da estrutura, com valores de peso bastante competitivos. A estrutura da Figura 37a obteve um peso quase idêntico ao mínimo encontrado, porém com um número maior de barras excluídas. Quando comparamos as estruturas encontradas com a geometria da estrutura inicial do problema (em azul), percebemos que a inclusão da otimização geométrica possibilitou que os nós se movimentassem para baixo, adotando um

perfil semicircular. Esse movimento acaba direcionando melhor o fluxo dos esforços nas barras, garantindo que o material utilizado na estrutura seja solicitado de forma otimizada.

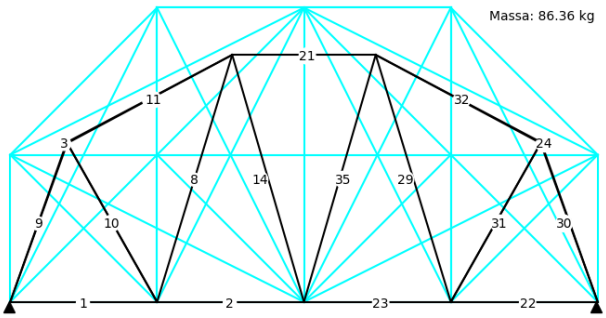
Figura 36 – Melhor estrutura encontrada para o problema 3b



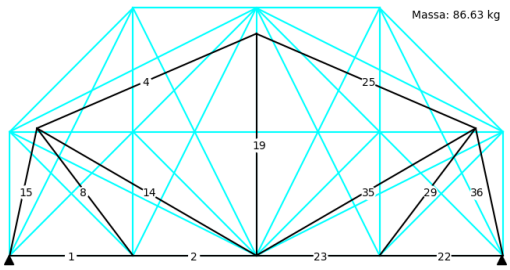
Fonte: Autor

Figura 37 – Estruturas alternativas encontradas para o problema 3b

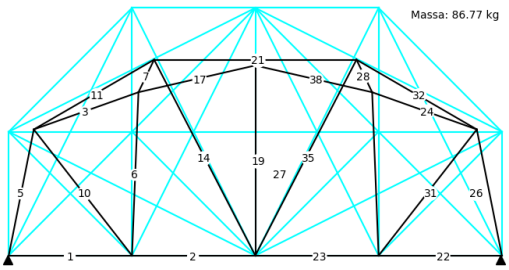
(a) 86,36 kg



(b) 86,63 kg



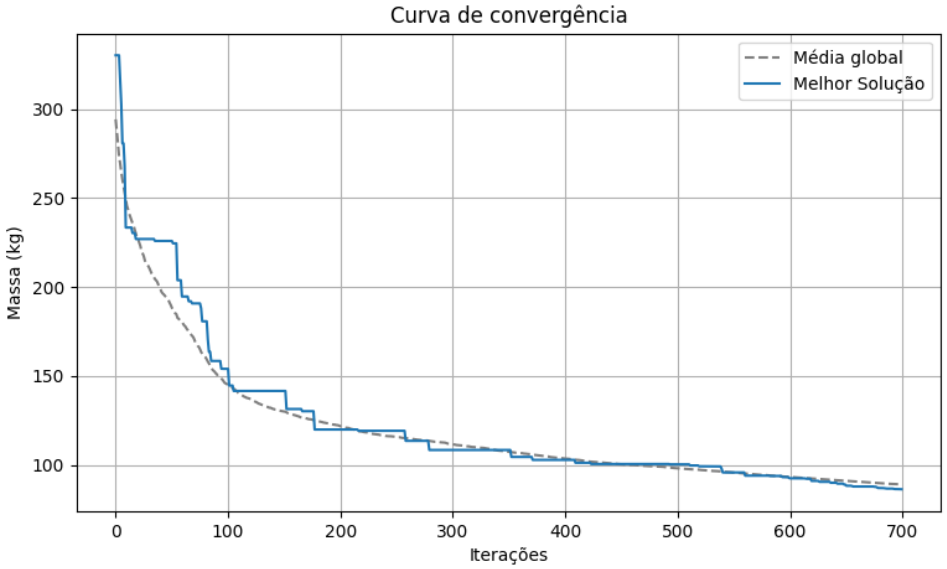
(c) 86,77 kg



Fonte: Autor

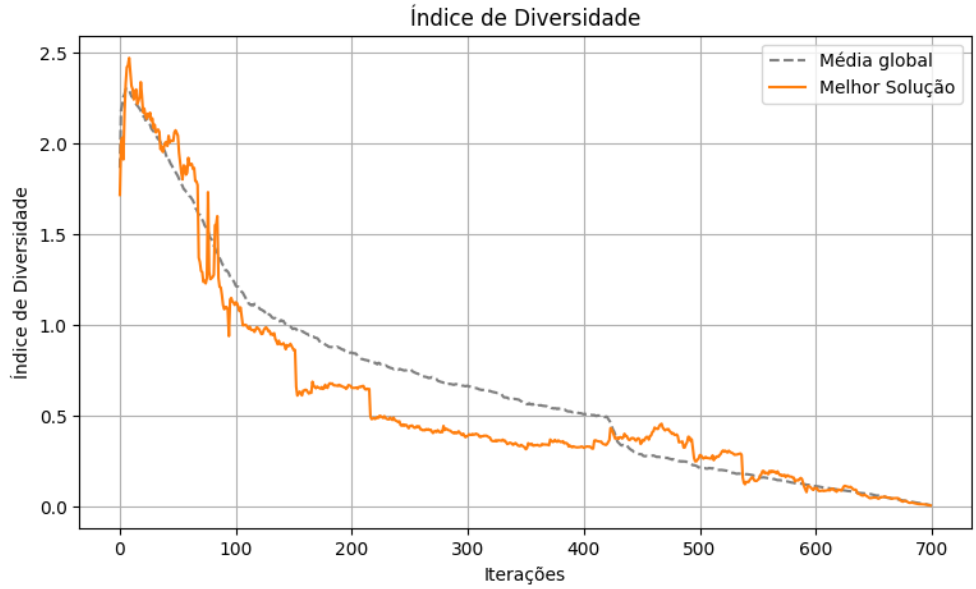
Por fim, as Figuras 38 e 39 apresentam a curva de convergência e o índice de diversidade do processo de otimização. No gráfico de índice de diversidade, a melhor estrutura possui um declínio mais intenso da diversidade no início da otimização. Já quando se observa a média global nesse gráfico, percebe-se um patamar formado na iteração 420, exatamente a iteração de mudança entre o processo de busca global e busca local.

Figura 38 – Curva de convergência para o problema 3b



Fonte: Autor

Figura 39 – Índice de Diversidade para o problema 3b



Fonte: Autor

7 CONCLUSÃO

De forma geral, o método de otimização por Search Group Algorithm (SGA) desenvolvido neste trabalho obteve bons resultados quando comparado a outros estudos em cima dos mesmos problemas. No problema 1, o método encontrou o melhor resultado na aplicação da otimização de uma treliça simples de 11 barras. De forma similar, no problema 3b - uma treliça de 39 barras com maior nível de complexidade - foi encontrado o melhor resultado entre os autores que utilizaram um campo amostral na avaliação dos resultados, embora outros autores tenham conseguido encontrar pesos menores. Já nos problemas 2b e 3a, foram obtidos resultados similares aos estudos comparados, obtendo um desempenho aceitável. Foi apenas no problema 2a que o algoritmo aplicado encontrou resultados menos satisfatórios (2,19% e 6,86% maiores que os autores comparados), não conseguindo convergir para mínimos de mesma ordem de grandeza.

Para o problema 2a, observou-se uma dificuldade na escolha de parâmetros corretos, sobretudo quanto à penalização proporcional aplicada às restrições. Trata-se de uma característica marcante na maioria dos algoritmos metaheurísticos de otimização: a alta dependência dos parâmetros para cada problema, que faz com que com que o operador do sistema tenha que adaptar os parâmetros (*parameter tunning*) para problemas específicos.

Com relação ao método de penalização aplicado, o refino do parâmetro $penal_{prop}$ mostrou-se bastante desafiador, já que o valor atribuído dependia diretamente da ordem de grandeza do mínimo global da estrutura. Assim, em cada problema aplicado foi necessária uma série de testes antes de encontrar o melhor valor para o parâmetro. Valores pequenos permitiam que estruturas inválidas surgissem como mínimos globais, valores muito altos impediam a exploração de áreas próximas aos limites do problema. Apesar dessa alta dependência do método de penalização aplicado, optou-se pela sua utilização em detrimento à penalização por pena de morte, pois esta última impedia a ampla exploração de soluções no domínio, eliminando sem qualquer diferenciação todas as estruturas que transpassassem a restrição, não importando o grau de transpasse. Ainda, uma série de métodos de penalização para metaheurísticos foram estudados por outros autores, inclusive métodos que independem do problema aplicado, de forma que é válido o estudo do impacto desses métodos aplicados ao algoritmo SGA.

Outro ponto avaliado por meio da comparação dos problemas foi o coeficiente de variação obtido com o uso do SGA, que se provou o menor em relação aos outros autores. Por mais que nem sempre o algoritmo encontrasse os menores mínimos entre os autores, esse ponto demonstra a consistência do SGA, retornando bons valores de forma independente da população inicial gerada, um dos princípios dos métodos de otimização por metaheurísticos.

Pode-se comentar ainda que o processo de desenvolvimento do código do Search Group Algorithm foi bem sucedido. Além da entrega de resultados consistentes e compe-

titivos, ele foi desenvolvido de tal forma que o mesmo algoritmo pode ser aplicado para diferentes problemas de otimização de treliças planas, envolvendo restrições de deslocamento, tensão admissível, flambagem de barras e validade da estrutura. Além disso, o algoritmo permite aplicar problemas envolvendo os três tipos simultâneos de otimização estrutural (dimensional, geométrica e topológica) e envolver variáveis contínuas, discretas ou mistas. Por fim, é possível lidar com estruturas simétricas, em que uma variável pode atribuir valores a um conjunto de barras e nós.

Dessa forma, tendo em vista os pontos comentados nesta seção e ao longo deste estudo, depreende-se que o presente trabalho conseguiu atingir todos os seus objetivos gerais e específicos ao trabalhar com a otimização estrutural por meio do algoritmo metaheurístico SGA, contribuindo com um tema cada vez mais importante no cenário atual da engenharia estrutural e da digitalização.

7.1 TRABALHOS FUTUROS

A partir dos resultados apresentados por esse trabalho de conclusão de curso e a fim de trazer novas contribuições para a otimização estrutural por meio de metaheurísticos, sugere-se a realização dos seguintes estudos:

- Utilização de diferentes tipos de penalização no algoritmo SGA e análise comparativa.
- Comparação da performance computacional do SGA aplicado nas linguagens MATLAB e Python.
- Aplicação do SGA para problemas de treliças espaciais.
- Consideração de efeitos de segunda ordem para os problemas aplicados neste estudo.
- Utilização de restrições relacionadas às Normas ABNT para estruturas metálicas aplicados ao SGA.

REFERÊNCIAS

AHRARI, Ali; ATAI, Ali A.; DEB, Kalyanmoy. Simultaneous topology, shape and size optimization of truss structures by fully stressed design based on evolution strategy. **Structural and Multidisciplinary Optimization**, v. 47, p. 1063–1084, 2014.

AZEEZ, Mohammed Nooruldeen; ALSAFFAR, Angham. Construction Time-Cost Optimization Modeling Using Ant Colony Optimization. **Journal of Engineering**, v. 20, 2014.

BALLING, Richard J.; BRIGGS, Ryan R.; GILLMAN, Kevin. Multiple Optimum Size/Shape/Topology Designs for Skeletal Structures Using a Genetic Algorithm. **Journal of Structural Engineering**, v. 132, p. 1158–1165, 2006.

BARBARESCO, Guilherme Marques. **Otimização de Problemas de Engenharia Utilizando o Algoritmo da Competição Imperialista (ICA)**. 2014. F. 73. Monografia (Trabalho de Conclusão de Curso) – Departamento de Engenharia Civil, Universidade Federal de Santa Catarina (UFSC), Florianópolis.

BAYRAM, Adem *et al.* Modeling stream dissolved oxygen concentration using teaching–learning based optimization algorithm. **Environ Earth Sci**, v. 73, p. 6565–6576, 2015.

CARRARO, Felipe. **Otimização estrutural de pórticos planos utilizando o algoritmo SGA**. 2015. F. 95. Monografia (Trabalho de Conclusão de Curso) – Departamento de Engenharia Civil, Universidade Federal de Santa Catarina (UFSC), Florianópolis.

CIVICIOGLU, Pinar. Backtracking Search Optimization Algorithm for numerical optimization problems. **Applied Mathematics and Computation**, v. 219, p. 8121–8144, 2013.

DEB, Kalyanmoy; GULATI, Surendra. Design of Truss-Structures for Minimum Weight using Genetic Algorithms. **Finite Elements in Analisys and Design**, v. 32, p. 1–18, 2000.

DEDE1, Tayfun *et al.* Usage of Optimization Techniques in Civil Engineering During the Last Two Decades. **Iris Publishers**, 2019.

ELBELTAGI, Emad *et al.* Overall multiobjective optimization of construction projects scheduling using particle swarm. **Engineering, Construction and Architectural Management**, v. 23, p. 265–282, 2016.

FAYAD, Hala; PERALTA, Richard C.; FORGHANI, Ali. Optimizing Reservoir-Stream-Aquifer Interactions for Conjunctive Use and Hydropower Production. **Advances in Civil Engineering**, p. 1–10, 2012.

FURTADO, Jean Jaques Howard Capristano. **Otimização de Longarinas de Pontes em Concreto Armado**. 2018. F. 111. Monografia (Trabalho de Conclusão de Curso) – Departamento de Engenharia Civil, Universidade Federal de Santa Catarina (UFSC), Florianópolis.

GANDOMI, Amir H.; ROKE, Ali R. Kashani David A.; MOUSAVI, Mehdi. Optimization of retaining wall design using recent swarm intelligence techniques. **Engineering Structures**, v. 103, p. 72–84, 2015.

GEEM, Zong Woo. **Studies in Computational Intelligence**: Harmony Search Algorithms for Structural Design Optimization. 239. ed. Indiana: [*s.n.*], 2009.

GEEM, Zong Woo; KIM, Joong Hoon. A New Heuristic Optimization Algorithm: Harmony Search. **Simulation**, v. 76, p. 60–68, 2001.

GLOVER, Fred. Future Paths for Integer Programming and Links to Artificial Inteligence. **Computer Operations**, v. 13, p. 533–549, 1986.

_____. Heuristics for integer programming using surrogate constraints. **Decision Sciences**, v. 8, p. 156–166, 1977.

GOLDBERG, David E. **Genetic Algorithm in Search, Optimization and Machine Learning**. 1. ed. Alabama: [*s.n.*], 1989.

GONÇALVES, Matheus S.; LOPEZ, Rafael H.; MIGUEL, Leandro F. F. Search group algorithm: A new metaheuristic method for the optimization of truss structures. **Computers and Structures Journal by Elsevier**, v. 153, p. 165–184, 2015.

GONZALEZ, Teofilo F. **Handbook of approximation algorithms and metaheuristics**. 1. ed. Santa Barbara: [*s.n.*], 2007.

HARSONO, K *et al.* Comparative Study of Particle Swarm Optimization Algorithms in Solving Size, Topology, and Shape Optimization. **Journal of Physics: Conference Series**, v. 1625, p. 012015, 2020.

HOLLAND, John H. **Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence**. 1. ed. Michigan: [*s.n.*], 1975.

KAVEH, A. **Applications of Metaheuristic Optimization Algorithms in Civil Engineering**. 1. ed. Tehran, Iran: [*s.n.*], 2017.

KAVEH, A.; GHAZAAN, M. Ilchi. Enhanced colliding bodies optimization for design problems with continuous and discrete variables. **Advances in Engineering Software**, v. 77, p. 66–75, 2014.

KAVEH, A.; ZOLGHADR, A. A new PSRO algorithm for frequency constraint truss shape and size optimization. **Structural Engineering Mechanics**, v. 52, p. 445–468, 2014.

KENNEDY, James; EBERHART, Russell. Particle Swarm Optimization. **IEEE Conference Publication**, p. 1942–1948, 1995.

KIRKPATRICK, S.; C. D. GELATT, Jr.; VECCHI, M. P. Optimization by Simulated Annealing. **Science**, v. 220, p. 671–680, 1983.

LIU, Yue; CHANG, Gang-Len. An arterial signal optimization model for intersections experiencing queue spillback and lane blockage. **Transportation Research Part C**, v. 19, p. 130–144, 2011.

LUH, Guan-Chun; LIN, Chun-Yi. Optimal design of truss structures using ant algorithm. **Structural and Multidisciplinary Optimization**, v. 36, p. 365–379, 2008.

_____. Optimal design of truss-structures using particle swarm optimization. **Computers Structures**, v. 89, p. 2221–2232, 2011.

MARCO DORIGO ALBERTO COLORNI, Vittorio Maniezzo. Distributed Optimization by Ant Colonies. **European Conference on Artificial Life 91**, p. 134–142, 1991.

MARTINI, Kirk. Harmony Search Method for Multimodal Size, Shape, and Topology Optimization of Structural Frameworks. **Journal of Structural Engineering**, v. 137, p. 1332–1339, 2011.

METROPOLIS, Nicholas *et al.* Equation of State Calculations by Fast Computer Machines. **The Journal of Chemical Physics**, v. 21, p. 1087–1092, 1953.

MEZURA-MONTES, Efrén; COELLO, Carlos A. Coello. Constraint-handling in nature-inspired numerical optimization: Past, present and future. **Swarm and Evolutionary Computation - Elsevier**, v. 1, p. 173–179, 2011.

MIGUEL, Leandro Fleck Fadel; LOPEZ, Rafael Holdorf; MIGUEL, Letícia Fleck Fadel. Multimodal size, shape, and topology optimisation of truss structures using the Firefly algorithm. **Advances in Engineering Software - Elsevier**, v. 56, p. 23–37, 2013.

RAJAN, S. D. SIZING, SHAPE, AND TOPOLOGY DESIGN OPTIMIZATION OF TRUSSES USING GENETIC ALGORITHM. **Journal of Structural Engineering**, v. 110, p. 1480–1487, 1995.

RAO, R.V.; SAVSANI, V.J.; VAKHARIA, D.P. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. **Computer-Aided Design**, v. 43, p. 303–315, 2011.

RAO, SINGIRESU S. **Engineering Optimization**: Theory and Practice. 3. ed. Indiana: [*s.n.*], 1996.

RIBEIRO, Luiz Augusto Diamante. **Otimização estrutural de treliças utilizando o algoritmo Firefly**. 2014. F. 114. Monografia (Trabalho de Conclusão de Curso) – Departamento de Engenharia Civil, Universidade Federal de Santa Catarina (UFSC), Florianópolis.

ROMAN, Rafael Roberto. **Otimização de Coberturas Metálicas de Edifícios Industriais**. 2016. F. 134. Monografia (Trabalho de Conclusão de Curso) – Departamento de Engenharia Civil, Universidade Federal de Santa Catarina (UFSC), Florianópolis.

SHARMA, Ramnik; KUMARI, Anita. An arterial signal optimization model for intersections experiencing queue spillback and lane blockage. **International Journal of Computer Science and Mobile Computing**, v. 4, p. 271–277, 2015.

SILVEIRA, Ricardo. **Aplicação do Firefly Algorithm para Otimização Estrutural de Parâmetros de Pórticos Planos Submetidos à Flexão Composta com Imposição de Restrições**. 2014. F. 79. Monografia (Trabalho de Conclusão de Curso) – Departamento de Engenharia Civil, Universidade Federal de Santa Catarina (UFSC), Florianópolis.

SOUZA, Rafael R. de; MIGUEL, Leandro F. Fadel; LOPEZ, Rafael H. A Backtracking Search Algorithm for the Simultaneous Size, Shape and Topology Optimization of Trusses. **Latin American Journal of Solids and Structures**, v. 13, p. 15, 2016.

SOUZA, Rafael R. de; MIGUEL, Leandro Fleck Fadel; LOPEZ, Rafael Holdorf *et al.* A procedure for the size, shape and topology optimization of transmission line tower structures. **Engineering Structures - Elsevier**, v. 111, p. 162–184, 2016.

VANDERPLAATS, G.N. Thirty years of modern structural optimization. **Advances in Engineering Software**, v. 16, p. 81–88, 1992.

WEAVER, William; GERE, James M. **Matrix Analysis of Framed Structures**. 3. ed. New York: [*s.n.*], 1990. Disponível em: `https://link.springer.com/chapter/10.1007/978-1-4684-7487-9_4`. Acesso em: 13 fev. 2022.

WU, Chun-Yin; TSENG, Ko-Ying. Truss structure optimization using adaptive multi-population differential evolution. **Structural and Multidisciplinary Optimization**, v. 42, p. 575–590, 2010.

WU, Shyue-Jian; CHOW, Pei-Tse. Integrated Discrete and Configuration Optimization of Trusses Using Genetic Algorithm. **Computer Structures - Elsevier**, v. 55, p. 695–702, 1995.

YAMAMOTO, Karina Assis Rocha. **Otimização estrutural de pórticos planos com o uso do algoritmo SGA e análise não-linear**. 2015. F. 120. Monografia (Trabalho de Conclusão de Curso) – Departamento de Engenharia Civil, Universidade Federal de Santa Catarina (UFSC), Florianópolis.

YANG, Xin-She. Firefly Algorithms for Multimodal Optimization. **Computer Sciences**, v. 5792, p. 169–178, 2009.

YEPES, Vıctor *et al.* A parametric study of optimum earth-retaining walls by simulated annealing. **Engineering Structures by Elsevier**, v. 30, p. 821–830, 2008.

APÊNDICE A – DETALHAMENTO DE VARIÁVEIS DE PROJETO

Nas tabelas abaixo foram detalhadas as variáveis de projeto utilizadas para cada problema, incluindo as variáveis resultantes deste estudo e dos autores comparados.

Tabela 21 – Resultados completos obtidos para o problema 1

Variáveis	**Rajan (1995)**	**Balling, Briggs e Gillman (2006)**	**Martini (2011)**	**Miguel, Lopez e Miguel (2013)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
y1 (m)	19.987	-	-	-	20.057	20.067
y2 (m)	14.084	-	-	-	12.384	12.359
y3 (m)	4.7371	-	-	-	-	-
Elementos						
2	74.193	-	-	74.193	74.193	74.193
3	23.226	-	-	18.581	18.581	18.581
4	74.193	-	-	37.032	37.032	37.032
5	63.871	-	-	74.193	74.193	74.193
6	9.677	-	-	46.581	46.581	46.581
8	67.097	-	-	-	-	-
9	77.419	-	-	87.097	87.097	87.097
10	60.645	-	-	-	-	-
OFE	3840	500000	4075	50000	50000	50000
Mínimo (kg)	1475.999	1241.029	1315.418	1227.040	1227.070	1227.040
Média (kg)	-	-	-	1268.28	1265.654	1273.394
C.O.V (%)	-	-	-	2.12	2.02	1.78

Fonte: Elaborada pelo Autor (2022)

Tabela 22 – Resultados completos obtidos para o problema 2a

Variáveis	**Wu e Chow (1995)**	**Miguel, Lopez e Miguel (2013)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
A1	38.4	7.574	7.574	6.974
A2	11.381	6.974	6.974	6.155
A3	9.6	2.839	1.742	1.742
A4	38.4	11.381	13.819	11.381
A5	13.819	9.6	8.6	9.6
A6	9.6	6.974	7.574	6.155
A7	18.064	0.716	0.716	1.123
A8	8.6	1.419	0.716	1.123
A9	2.239	9.6	1.852	2.239
A10	6.974	1.123	0.91	2.839
A11	8.6	1.419	0.716	2.839
A12	6.974	1.742	1.123	1.742
A13	8.6	8.6	8.6	9.6
A14	11.381	1.852	2.839	2.239
A15	11.381	8.6	6.155	6.155
X2	301.98314	278.27986	334.33883	288.33590
X2	565.32526	571.15964	590.35225	604.9928
Y2	274.3708	262.41248	254	254.31837
Y3	315.13272	255.1049	254	254
Y4	139.2809	131.00304	132.36646	141.04004
Y6	0.88138	43.3324	50.8	9.9064
Y7	39.41064	48.31588	50.8	18.20120
Y8	88.16594	124.06884	132.1094	82.96692
OFE	-	8000	8000	8000
Peso mínimo (kg)	182.506	62.627	59.641	64.032
Peso médio (kg)	-	69.948	64.049	69.856
C.O.V (%)	-	3.85	3.96	2.87

Fonte: Elaborada pelo Autor (2022)

Tabela 23 – Resultados completos obtidos para o problema 2b

Variáveis	**Miguel, Lopez e Miguel (2013)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
A1	6.155	6.155	6.155
A2	6.155	6.155	6.155
A3	0.716	6.155	0
A4	13.819	13.819	13.819
A5	6.794	8.6	6.974
A6	8.6	0.91	8.6
A7	0.716	1.742	0.91
A8	1.419	3.478	1.852
A9	7.574	2.839	1.419
A10	2.839	2.839	2.839
A11	0	0	0
A12	0	1.742	0
A13	11.381	0	9.6
A14	3.477	0	3.478
A15	0	11.381	1.419
X2	335.26984	322.4111	341.8650
X3	594.3219	561.5408	616.1154
Y2	293.11854	254	275.2593
Y3	296.89552	254	258.3290
Y4	134.7851	139.9969	142.6556
Y6	35.39998	-9.4016	18.7406
Y7	29.4767	50.8	17.3369
Y8	131.8133	138.7773	129.9905
OFE	8000	8000	8000
Peso mínimo (kg)	56.803	54.827	56.016
Peso médio (kg)	69.223	60.187	66.973
C.O.V (%)	8.69	7.88	6.01

Fonte: Elaborada pelo Autor (2022)

Tabela 24 – Resultados completos obtidos para o problema 3a

Variáveis	**Deb e Gulati (2000)**	**Luh e Lin (2008)**	**Wu e Tseng (2010)**	**Luh e Lin (2011)**	**Miguel, Lopez e Miguel (2013)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
1	-	0.329	0.323	0.252	0.323	0.323	0.335
2	4.845	4.845	4.839	4.839	4.854	5.152	4.865
3	0.329	-	-	-	-	-	-
5	9.69	9.690	9.677	9.677	9.678	9.678	9.672
7	0.335	-	-	-	-	-	-
8	1.619	1.613	1.613	1.613	1.615	1.3	1.616
9	0.329	-	-	-	-	-	-
10	6.845	6.852	6.839	6.845	6.869	7.285	6.839
11	6.858	6.858	6.839	6.845	6.846	5.516	6.842
14	3.606	3.613	3.606	3.613	3.615	2.907	3.606
16	-	-	-	-	-	1.399	-
19	-	-	-	-	-	1.251	-
21	6.484	6.452	6.452	6.452	6.452	5.201	6.478
22	-	0.329	0.323	0.252	0.323	0.323	0.335
23	1.619	1.613	1.613	1.613	1.615	1.3	4.865
25	0.329	-	-	-	-	-	-
26	9.69	9.690	9.677	9.677	9.678	9.678	9.671
28	0.335	-	-	-	-	-	-
29	1.619	1.613	1.613	1.613	1.615	1.3	1.616
30	0.329	-	-	-	-	-	-
31	6.845	6.852	6.839	6.845	6.869	7.285	6.839
32	6.858	6.858	6.839	6.845	6.846	5.516	6.842
35	3.606	3.613	3.606	3.613	3.615	2.907	3.606
37	-	-	-	-	-	1.399	-
OFE	504000	303600	32300	262500	50000	50000	50000
Peso mínimo (kg)	89.152	87.758	87.634	87.549	87.792	87.637	87.728
Peso médio (kg)	-	-	-	-	100.552	93.284	89.199
C.O.V (%)	-	-	-	-	12.9	5.28	1.82

Fonte: Elaborada pelo Autor (2022)

Tabela 25 – Resultados completos obtidos para o problema 3b

Variáveis	**Deb e Gulati (2000)**	**Luh e Lin (2008)**	**Wu e Tseng (2010)**	**Luh e Lin (2011)**	**Miguel, Lopez e Miguel (2013)**	**Ahrari, Atai e Deb (2014)**	**Souza, Miguel e Lopez (2016)**	**Este estudo**
1	3.839	2.110	1.052	2.090	1.901	0.596102	2.879	2.615
2	7.523	7.065	9.735	7.058	6.87	6.321323	6.435	7.069
3	-	-	-	-	-	7.844571	2.575	2.196
4	-	-	-	-	7.686	7.110805	-	0.848
5	10.419	9.923	5.774	9.903	-	9.695815	10.112	8.939
6	-	-	-	-	-	3.610361	1.259	0.3399
7	-	-	-	-	-	6.779051	0.336	5.229
8	0.329	0.523	1.097	-	8.1	-	-	2.873
10	7.452	7.877	7.245	7.877	-	5.310751	6.089	4.889
11	3.252	8.123	7.323	8.116	-	0.322935	5.483	5.508
12	-	-	-	0.529	-	-	3.437	-
13	-	-	-	-	-	-	-	1.236
14	8.342	3.387	3.503	3.387	0.326	2.334415	-	0.3299
15	-	-	-	-	9.887	-	-	1.283
17	-	-	-	-	-	-	-	7.334
19	-	-	-	-	6.200	2.009196	-	4.249
20	-	-	-	-	-	-	2.29	-
21	8.671	8.103	7.135	8.097	-	-	5.351	-
22	3.839	2.110	1.052	2.090	1.901	0.596102	2.879	2.615
23	7.523	7.065	9.735	7.058	6.87	6.321323	6.435	7.069
24	-	-	-	-	-	7.844571	2.575	2.196
25	-	-	-	-	7.686	7.110805	-	0.848
26	10.419	9.923	5.774	9.903	-	9.695815	10.112	8.939
27	-	-	-	-	-	3.610361	1.259	0.3399
28	-	-	-	-	-	6.779051	0.336	5.229
29	0.329	0.523	1.097	-	8.1	-	-	2.873
31	7.452	7.877	7.245	7.877	-	5.310751	6.089	4.889
32	3.252	8.123	7.323	8.116	-	0.322935	5.483	5.508
33	-	-	-	0.529	-	-	3.437	-
34	-	-	-	-	-	-	-	1.236
35	8.342	3.387	3.503	3.387	0.326	2.334415	-	0.3299
36	-	-	-	-	9.887	-	-	1.283
38	-	-	-	-	9.887	-	-	7.334
OFE	504000	453600	137200	262500	50000	40256	50000	50000
Peso mínimo (kg)	87.176	85.607	85.469	85.541	86.774	82.091	86.753	86.356
Peso médio (kg)	-	-	-	-	94.478	-	100.59	89.095
C.O.V (%)	-	-	-	-	5.3	-	4.6	2.8

Fonte: Elaborada pelo Autor (2022)