**CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS**
**CAMPUS TIMÓTEO**

Fernanda Silva Dias

# ANÁLISE DA CONVERGÊNCIA E MANUTENÇÃO DA DIVERSIDADE DE UMA NOVA FUNÇÃO TESTE UTILIZANDO PROBLEMAS DE MUITOS OBJETIVOS

**Timóteo**

**2018**

**Fernanda Silva Dias**

# ANÁLISE DA CONVERGÊNCIA E MANUTENÇÃO DA DIVERSIDADE DE UMA NOVA FUNÇÃO TESTE UTILIZANDO PROBLEMAS DE MUITOS OBJETIVOS

Monografia apresentada à Coordenação de Engenharia de Computação do Campus Timóteo do Centro Federal de Educação Tecnológica de Minas Gerais para obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Douglas Nunes de Oliveira

Timóteo

2018

Fernanda Silva Dias

**Análise da convergência e manutenção da diversidade de uma nova função teste utilizando problemas de muitos objetivos**

Monografia apresentada à Coordenação de Engenharia de Computação do Campus Timóteo do Centro Federal de Educação Tecnológica de Minas Gerais para obtenção do grau de Bacharel em Engenharia de Computação.

Trabalho aprovado, 10 de dezembro de 2018:

Douglas Nunes de Oliveira
Orientador

João Batista Queiroz Zuliani
Professor convidado

Luciano Nascimento Moreira
Professor convidado

Timóteo
2018

Dedico este trabalho aos meus pais, pelo constante incentivo em meu crescimento profissional e pessoal.

# Agradecimentos

Agradeço primeiramente a Deus, por me conceder saúde e sabedoria para enfrentar as dificuldades e alcançar os meus objetivos.

Aos meus pais Luiz e Mônica, agradeço pelo apoio, incentivo e contribuição para a minha formação e crescimento pessoal e profissional.

Aos meus familiares, agradeço os momentos de alegria e presença em minha vida!

Ao Diogo, pelo carinho e exemplos de dedicação e competência.

Ao orientador deste trabalho, Douglas Nunes de Oliveira, pela atenção a mim dispensada, boa vontade em ensinar e paciência no desenvolvimento deste trabalho.

A todos os professores do CEFET-MG, que contribuíram direta ou indiretamente para a minha formação, em especial, a professora Deisymar pelo empenho e orientações no desenvolvimento deste trabalho.

A todos os meus amigos do CEFET-MG, pelo verdadeiro trabalho em equipe, pela união nos momentos difíceis e pelos ensinamentos recebidos... sem vocês essa trajetória teria sido muito mais difícil!

“A menos que modifiquemos a nossa maneira de pensar, não seremos capazes de resolver os problemas causados pela forma como nos acostumamos a ver o mundo”. (Albert Einstein)

# Resumo

A otimização está presente em processos de tomada de decisão em diversas áreas. A comunidade de computação evolucionária estuda e propõe algoritmos de otimização eficientes para resolver problemas de múltiplos objetivos em diferentes cenários. Esses problemas, também conhecidos como funções de teste, são o ponto chave para impulsionar novos algoritmos e avaliar os já existentes. As funções de teste podem apresentar diferentes características que aumentam sua complexidade e impulsionam a melhoria dos algoritmos de muitos objetivos, de forma a lidarem melhor com problemas cada vez mais próximos do mundo real. Este trabalho propôs uma nova função de teste chamada DNO$_f$ e analisou sua viabilidade ao ser otimizada pelo algoritmo NSGA-III para até 15 objetivos. O mesmo processo foi executado para a função DTLZ1. Os resultados finais foram obtidos através da comparação da métrica de desempenho *Inverted Generational Distance* (IGD) para as duas funções. A função DNO$_f$ proposta apresentou maior nível de dificuldade ao ser otimizada pelo NSGA-III em todos os objetivos.

**Palavras-chave**:NSGA-III, computação evolucionária, funções teste.

# Abstract

Optimization is present in decision-making processes in several fields. The evolutionary computing community studies and proposes efficient optimization algorithms to solve problems of multiple objectives in different scenarios. These problems, also known as test functions, are the key to boosting new algorithms and evaluating existing ones. The test functions can have different characteristics that increase their complexity and drive the improvement of many-objective optimization algorithms, in order to better deal with problems that are closer to the real world. This work proposed a new test function called DNO$_f$ and analyzed its viability when optimized by the NSGA-III algorithm for up to 15 objectives. The same process was performed for the DTLZ1 function. The final results were obtained by comparing the Inverted Generational Distance (IGD) performance metric for the two functions. The proposed DNO$_f$ function presented higher level of difficulty when optimized by NSGA-III in all the objectives.

**Keywords**: NSGA-III, evolutionary computing, test functions.

# Lista de ilustrações

# Lista de tabelas

# Sumário

# 1 Introdução

A otimização está presente em processos de tomada de decisão e na análise de sistemas físicos, da engenharia à ciência e da economia à ciências sociais. Pode-se afirmar que a otimização é um procedimento para encontrar a melhor solução possível, uma vez que um problema tenha sido formulado. Esta solução pode um conjunto de valores ótimos para um problema de minimização ou maximização (DEB, 2001), (RAO; RAO, 2009), (NOCEDAL; WRIGHT, 1999).

A comunidade de computação evolucionária tem demonstrado um interesse significativo na otimização por muitos anos, em particular à resolução de problemas "caixa preta", ao qual métodos exatos e analíticos não se aplicam. A popularidade de ferramentas computacionais disponíveis incentivou o aumento de aplicações de técnicas de otimização a problemas reais, portanto muitos algoritmos, em particular os evolucionários, foram propostos para resolução dessa classe de problemas, como por exemplo: Differential Evolution (DE) (STORN; PRICE, 1997), Nondominated Sorting Genetic Algorithm II (NSGA-II) (DEB et al., 2002), entre outros (LOBATO et al., 2008), (VESTERSTROM; THOMSEN, 2004).

Problemas conhecidos como *single-objective* são resolvidos com a tentativa de obter o melhor valor ótimo que resolva o problema, sendo este o mínimo global ou o máximo global. Já em um cenário multiobjetivo, onde há dois ou três objetivos e em um cenário *many-objective*, onde há quatro ou mais objetivos, não há uma solução que seja ótima para todos os objetivos, e sim um conjunto de soluções que seja superior a todo o restante (SRINIVAS; DEB, 1994).

Justesen (2009) afirma que algoritmos evolutivos são otimizadores inspirados na evolução darwiniana e, com isso, o conceito de sobrevivência do mais apto. Muitos algoritmos evolutivos foram propostos para obter boas soluções para problemas de dois ou mais objetivos como: SPEA2 (ZITZLER; LAUMANNS; THIELE, 2001), NSGA-II (DEB et al., 2002), MOEA/D (ZHANG; LI, 2007), NSGA-III (DEB; JAIN, 2014) e MOEA/DD (LI et al., 2015).

De acordo com Vesterstrom e Thomsen (2004), muitos esforços foram dedicados para comparar esses algoritmos entre si, baseando essas comparações em problemas numéricos de *benchmark*, também tratado neste trabalho como "funções de teste", ou apenas "problemas".

Nametala, Pappa e Carrano (2017) afirmam que, medidas escalonáveis para avaliação de algoritmos multiobjetivos são importantes, pois permitem testa-los sob a mesma perspectiva. Dizer que um problema de teste é escalonável significa dizer que ele permite a utilização de M objetivos sem modificar a representação matemática do problema, como por exemplo o conjunto de problemas DTLZ propostos por (DEB et al., 2001), que dispôs inicialmente de nove funções de teste amplamente utilizadas para testes na academia.

Estudos sobre a busca de características que causam dificuldades a um algoritmo podem parecer trabalhos pessimistas, mas a verdadeira eficiência de um algoritmo é revelada

quando a ele se aplica problemas de teste difíceis e desafiadores (DEB, 1999).

## 1.1 Justificativa

Ao avaliar o algoritmo, é possível identificar o tipo de problema que ele domina, caso ele seja avaliado sob um conjunto de funções de teste. A importância de se criar novas funções de teste pode ser explicada a partir da necessidade de se testar novos algoritmos de otimização que irão atender aos problemas do mundo real cada vez mais complexos. (CHO et al., 2017)

É importante que mais propriedades possam ser acrescentadas em novas funções de teste garantindo sua dinamicidade, contribuindo para uma melhor avaliação de desempenho ou eficácia de um algoritmo de otimização em relação a outros. Dentre essas propriedades destacam-se a unimodalidade, multimodalidade, regularidade, irregularidade, separabilidade e multidimensionalidade. Essas funções podem incluir um ou mais ótimos globais e locais e podem ser manipuladas e modificadas para testar algoritmos de otimização em vários cenários (CHO et al., 2017) (JAMIL; YANG, 2013).

Este trabalho propõe a análise de uma nova função de teste chamada DNO$_f$, aplicando-a ao algoritmo evolutivo NSGA-III, podendo posteriormente contribuir para a avaliação de outros algoritmos evolutivos sob diferentes perspectivas e consequentemente estimular novas propostas de algoritmos que resolvam problemas mais complexos, podendo gerar como resultados novas métricas comparativas entre eles.

Mediante este cenário, levantou-se a seguinte questão de pesquisa: A nova função proposta apresentará um nível mais elevado de dificuldade na convergência e manutenção da diversidade na soluções encontradas pelo algoritmo NSGA-III?

## 1.2 Objetivos

O objetivo geral deste trabalho é propor e avaliar uma nova função de teste chamada DNO$_f$, utilizando o algoritmo NSGA-III (DEB; JAIN, 2014) presente no *framework* Jmetal4 para executar a otimização da função e obter resultados através do índice de avaliação *Inverted Generational Distance* (IGD). O mesmo processo será realizado para a função DTLZ1 (DEB et al., 2001), de forma a obter resultados comparativos entre as duas funções.

# 2 Procedimentos Metodológicos

A presente pesquisa apresentou os seguintes passos para atingir os objetivos propostos:

1. Realizar o levantamento de artigos dos principais autores da computação evolucionária e adquirir um melhor entendimento do assunto abordado bem como a natureza das funções de teste;
2. Estudar o algoritmo NSGA-III, bem como a função de teste DTLZ1 e conceitos de álgebra linear, de forma a auxiliar o entendimento da função tratada neste trabalho;
3. Estudar o índice de avaliação de algoritmos *Inverted Generational Distance* (IGD) de forma a utilizá-lo para avaliar o algoritmo NSGA-III ao otimizar a função $\text{DNO}_f$ e DTLZ1;
4. Avaliar a nova função de teste proposta neste trabalho, através do algoritmo de otimização de muitos objetivos NSGA-III com auxílio do JMetal;
5. Estudar a real contribuição deste trabalho através da análise dos resultados do IGD das soluções obtidas pelo NSGA-III para as funções $\text{DNO}_f$ e DTLZ1.

# 3 Revisão Bibliográfica

O principal objetivo deste capítulo é apresentar os fundamentos teóricos para compreensão desse trabalho e de sua relevância.

## 3.1 Estado da Arte

Nessa seção serão citados artigos relevantes que contribuíram para a realização deste trabalho, especialmente na criação de funções de testes para algoritmos de muitos objetivos.

Deb et al. (2001) sugeriu três abordagens diferentes para projetar funções de teste com a finalidade de avaliar e comparar o desempenho de novos algoritmos multiobjetivos e também compreender seu funcionamento ao tratar problemas com mais de dois objetivos. Dentre essas abordagens incluem a simplicidade de construção, conhecimento da forma exata da Fronteira de Pareto, introdução de dificuldades mantendo um conjunto distribuído de soluções. Os autores criaram uma suíte de funções DTLZ[1-9] abordando as características citadas acima.

Em Jamil e Yang (2013) foi feita uma compilação de 175 funções de teste, de forma a facilitar a validação de novos algoritmos de otimização. Além disso, foram levantadas as características principais para a criação de uma função de testes. O objetivo dos autores através da compilação de funções foi encontrar os aspectos as tornam mais complexas, consequentemente deixando o processo de otimização mais difícil. As funções foram classificadas conforme suas características, dentre elas a multimodalidade, separabilidade, *valley landscape*, dimensionalidade, escalabilidade, entre outros.

Em Bradstreet et al. (2007) foi analisada a interação entre o conjunto de funções WFG proposta por Huband et al. (2005), aplicada aos algoritmos de otimização multiobjetivo (MOEAs) NSGA-II e SPEA2. Através da alteração dos parâmetros das funções do WFG, os autores exploraram diferentes problemas e as diferentes formas de abordagem dos algoritmos para resolvê-los. O objetivo final não era fazer uma comparação de qual algoritmo é superior, mas mostrar as características únicas do conjunto WFG para testar interações específicas entre os problemas e os MOEAs, permitindo que o usuário facilmente modifique as características necessárias e observe seus efeitos nos MOEAs.

## 3.2 Otimização Multiobjetivo

Problemas do mundo real geralmente têm muitos objetivos conflitantes e raramente existe uma solução que seja ótima para todos os objetivos simultaneamente. Esses problemas fazem parte de um estudo na otimização chamado Problemas de Otimização Multiobjetivo (MOPs). (BUI; ALAM, 2008) (SAMPAIO, 2011)

Dentre algumas definições encontradas na literatura, Justesen (2009) afirma que o principal conceito da otimização multiobjetivo consiste em um problema contendo várias fun-

ções a serem maximizadas ou minimizadas por uma solução x satisfazendo diferentes restrições. Já o autor Osyczka (1985) resume estes conceitos como um problema para encontrar um vetor de variáveis de decisão que satisfaça as restrições ao mesmo tempo otimizando um vetor de funções objetivo conflitando umas com as outras.

Justesen (2009), afirma que a transição de problemas de um único objetivo para múltiplos objetivos apresenta um desafio por se tratar no último caso de um vetor multidimensional ao invés de apenas um valor escalar.

Para resolver problemas multiobjetivos, Barboza (2011) diz que fazer a modelagem apropriada é o passo mais importante. Caso o modelo seja muito simples, não fornecerá as informações necessárias sobre o problema, e se ele for muito complexo sua solução poderá se tornar inviável.

Para realizar a modelagem do problema, é necessário identificar as funções objetivo, variáveis de decisão e as restrições do problema (WRIGHT; NOCEDAL, 1999). A função objetivo depende de um conjunto de variáveis de decisão que devem satisfazer as restrições e requisitos de modo a gerar uma solução viável. A partir do momento que a modelagem é formulada, o processo de otimização irá encontrar a solução que maximiza ou minimiza a função objetivo e satisfaça as restrições (RAO; RAO, 2009).

### 3.2.1 Dominância

Dizemos que uma solução **A** domina **B** caso **A** não seja pior que **B** em todos os objetivos e **A** seja melhor que **B** em pelo menos um dos objetivos. Se **A** não domina **B** e nem **B** domina **A**, dizemos que estes são pontos não dominados entre si (JUSTESEN, 2009) (BUI; ALAM, 2008).

Dado um conjunto de pontos no espaço dos objetivos, como por exemplo na Figura 1, é possível dizer se um ponto domina ou não o outro. Todos os pontos não dominados por outros pontos correspondem à primeira fronteira, que será detalhada no tópico Pareto Ótima (DEB, 2011).

Percebe-se que existem vários pontos em uma fronteira de pontos não dominados e que em cada um deles há um ganho em um objetivo e uma perda em outro. De acordo com Deb (2011), um ganho em um objetivo acontece apenas devido ao sacrifício em pelo menos um outro objetivo. Esse compromisso entre os objetivos traz um interesse para o usuário em encontrar uma variedade ampla de soluções antes de tomar uma decisão final.

### 3.2.2 Pareto Ótima

A Pareto ótima leva o nome do economista Italiano Vilfredo Pareto sendo uma medida de qualidade amplamente aceita em problemas de múltiplos objetivos (SEKULSKI, 2010).

O conceito de soluções ótimas foi originalmente proposto por Edgeworth (1881) e em seguida generalizado por Pareto (1896), portanto daí originou-se o termo "Pareto ótima", correspondendo ao conjunto de soluções não dominadas que atendem aos objetivos e restrições previamente propostas. As soluções deste conjunto formam uma superfície chamada de

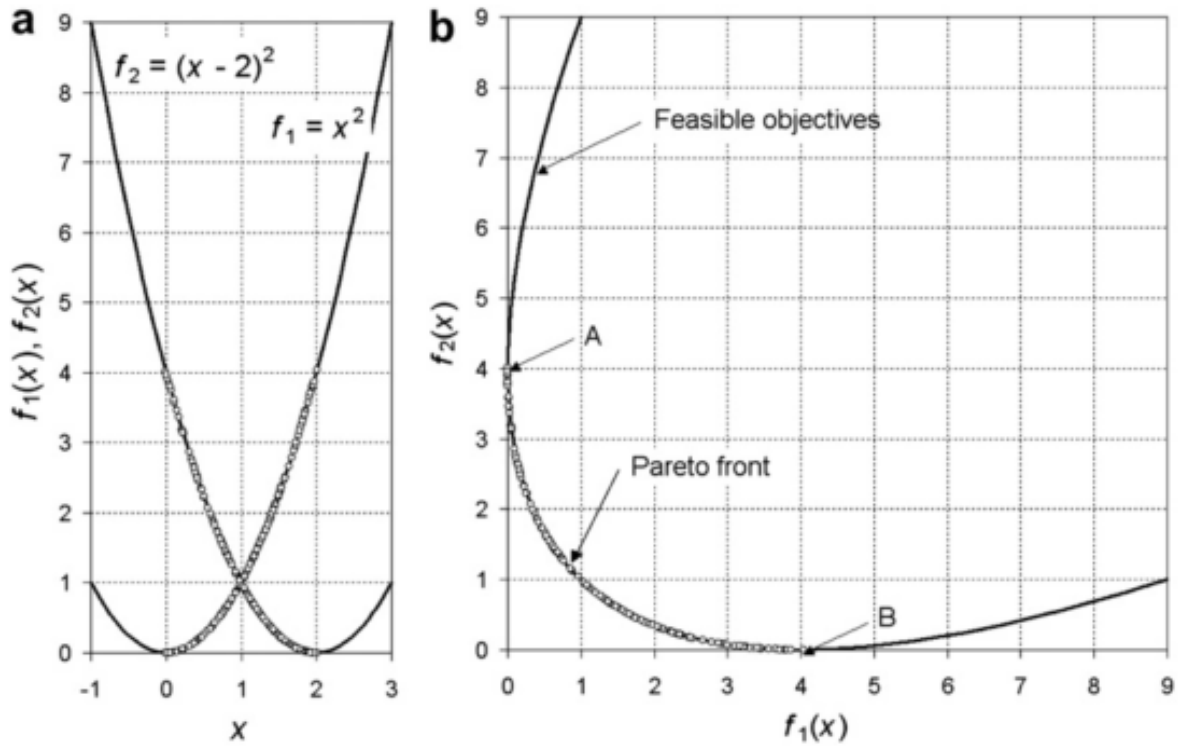
Fronteira de Pareto (ou Frente de Pareto) conforme demonstra a Figura 1 (COELLO, 2006) (FIGUEIREDO, 2013).

O conjunto solução gerado pelo algoritmo de otimização multiobjetivo possui algumas características a serem consideradas, que influenciam diretamente na qualidade das soluções encontradas na fronteira de Pareto. (FLEMING; PURSHOUSE; LYGOE, 2005)

São elas:

1. *Proximidade*. O conjunto de aproximação deve conter soluções cujo vetores objetivo sejam de valor aproximado ao da fronteira de Pareto.
2. *Diversidade*. O conjunto de aproximação deve conter uma boa distribuição de soluções diversificadas. O conjunto de pontos não-dominados deve estender ao longo do intervalo completo da fronteira de Pareto.

Figura 1 – Representação gráfica da função de teste Schaffer: (a) conjunto de soluções não dominadas no espaço da solução, (b) conjunto de objetivos não dominados no espaço objetivo.




Fonte: Adaptado de Sekulski (2010) e Figueiredo (2013)

O gráfico **a** representado na Figura 1 demonstra duas funções $f_1(x)$ e $f_2(x)$ com valores mínimos iguais a 0 e 2 respectivamente. Portanto, no intervalo [0,2] as duas funções estão em conflito, pois, para minimizar a função $f_1(x)$ é necessário aumentar o valor de $f_2(x)$, levando em consideração que não existe um único valor ótimo que minimize ambas simultaneamente (FIGUEIREDO, 2013).

O gráfico **b** representado na Figura 1, representa o percurso das funções no plano chamado espaço dos objetivos. O arco entre os pontos A e B representa o conjunto de soluções não dominadas, também chamado de fronteira de Pareto, sendo esse o conjunto de pontos de melhor compromisso entre os objetivos e a solução para esse problema de otimização. (FIGUEIREDO, 2013) (SEKULSKI, 2010)

## 3.3 Problemas de Otimização Multiobjetivo

Um problema de otimização multiobjetivo (MOP) padrão, correspondente ao termo em inglês (*MultiObjective Optimization Problem*), inclui um conjunto de $n$ parâmetros (variáveis de decisão), um conjunto de restrições $m$ e um conjunto $k$ de funções objetivas (ZITZLER, 1999).

A formação de um MOP generalizado é definida abaixo:

$$Minimizar/MaximizarF(x) = [f_1(x), f_2(x), ..., f_k(x)], \tag{3.1}$$

sujeito a

$$g(x) = [g_1(x), g_2(x), ..., g_q(x)] \leq 0, \tag{3.2}$$

onde cada $f_i$ $(i = 1, ..., k)$, define uma função objetivo e $g_i$ $(i = 1, ..., q)$ são as restrições que modelam o problema.

## 3.4 Problemas de Muitos Objetivos *(Many-Objective)*

Problemas *Many-Objective* são definidos como problemas que contem quatro ou mais objetivos. Esse tipo de problema gera uma grande dificuldade de visualização gráfica, pois a visualização acima de três dimensões não é clara. Geralmente estuda-se problemas de até 15 objetivos.

De acordo com Deb e Jain (2014), dentre as dificuldades ao se lidar com problemas de muitos objetivos podem-se destacar:

1. Uma boa parte da população é não-dominada: De acordo com Deb (1999), Poloni (1995), o aumento do número de objetivos influencia diretamente no crescimento da população não-dominada e dessa forma não sobra espaço para se criar uma nova solução em uma geração, dificultando o processo de busca e deixando o algoritmo evolucionário ineficiente.

2. Cálculo computacionalmente caro: avaliar a diversidade de soluções torna-se computacionalmente caro, especialmente em um cenário com alto número de dimensões.

3. Operação de recombinação pode se tornar ineficiente: Em um espaço de muitas dimensões, uma certa quantidade de soluções provavelmente estarão muito distantes umas das outras, dessa forma, o efeito do operador de recombinação poderá ser questionável pois irá produzir descendentes também distantes uns dos outros. Òutros operadores de

recombinação especiais poderão ser necessários para lidar eficientemente neste cenário.

4. Visualização das soluções: As soluções em um cenário de muitas dimensões se tornam difíceis de serem visualizadas, e requer outro tipo de ferramenta que a converta para um cenário de 2 ou 3 dimensões.

## 3.5 Algoritmos Evolucionários (AE's)

Algoritmos Evolucionários são algoritmos de otimização baseados na teoria da evolução e nos mecanismos de seleção natural, tais como a reprodução, mutação e seleção, tendo como objetivo principal a simulação do processo de evolução das espécies no computador para resolver problemas e encontrar a solução ótima (MACHADO, 2005) (COELLO; LAMONT, 2004).

### 3.5.1 Algoritmos Evolucionários de Muitos Objetivos (MaOEAs)

Os MaOEAs, também conhecidos como *Many-Objective Evolutionary Algorithms*, apresentam como resultado do cálculo dos valores objetivos, uma população de soluções em cada geração. A próxima geração será composta pelos melhores indivíduos da população anterior e seu resultado final após $n$ gerações será uma população de soluções ótimas aproximadas da Pareto ótima (COELLO; LAMONT, 2004) (DEB, 2001) (BUI; ALAM, 2008).

O objetivo de um MaOEA é alcançar a convergência e a diversidade do vetor de soluções não dominadas, em direção à fronteira de Pareto (YUAN; XU; WANG, 2014).

Dentre os algoritmos evolucionários de muitos objetivos existentes, foi escolhido o NSGA-III proposto por Deb e Jain (2014), para ser utilizado como objeto de pesquisa neste trabalho devido a sua eficiência em tratar problemas de muitos objetivos e também por ser um MaOEA proposto recentemente na literatura.

## 3.6 Algoritmo *Non-Dominated Sorting Genetic Algorithm III* (NSGA-III)

A motivação para o surgimento do algoritmo NSGA-III, foi o questionamento se a otimização evolucionária multiobjetiva é de fato eficiente ao tratar problemas com quatro ou mais objetivos (*many-objective*), isso devido à dificuldade dos algoritmos multiobjetivos encontrarem um número adequado de novas soluções para população à medida que o número de objetivos cresce.

O crescimento do número de objetivos faz aumentar a quantidade de soluções não-dominadas em uma população, impedindo que haja novas soluções em uma geração. Portanto, os algoritmos multiobjetivo ao ser executado em um cenário *many-objective* eram ineficientes (DEB; JAIN, 2014).

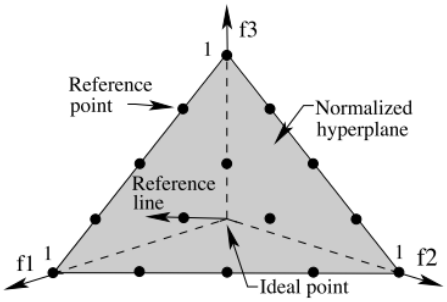
O NSGA-III permanece similar ao NSGA-II (DEB et al., 2002) porém com mudanças no mecanismo de seleção, utilizando pontos de referência igualmente distribuídos no espaço dos objetivos de forma a proporcionar a diversidade de soluções ótimas (DEB; JAIN, 2014).

A quantidade de pontos de referência a ser utilizada é definida pela equação 3.3 onde $A = m + p - 1$ e $p$ é a quantidade de subdivisões por eixo, $m$ a quantidade de objetivos (SCHEFFÉ, 1958), (RIBEIRO, 2016b), (RIBEIRO, 2018).

$$H = \left( \frac{A!}{p!(A-p)!} \right) \tag{3.3}$$

O procedimento de geração de pontos de referência é baseado especificamente na abordagem proposta por Das e Dennis (1998). Um exemplo deste procedimento de distribuição de tais pontos, com $M = 3$ e $p = 4$, que são o número de objetivos e o número de divisões respectivamente, é mostrado na figura 2.

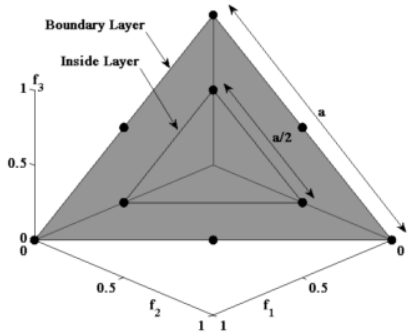
Figura 2 – 15 pontos de referência para problemas de 3 objetivos com p = 4.




Fonte: (DEB; JAIN, 2014)

Em um cenário onde há um alto número de objetivos, de forma a poupar a grande quantidade de pontos de referência, Deb e Jain (2014) propuseram a utilização de duas camadas de pontos de referência com valores pequenos para $p$, uma camada mais externa é chamada de *Boundary Layer* e a outra camada interna é chamada de *Inside Layer*. A geração destas duas camadas é uma extensão do procedimento definido por Das e Dennis (1998) e representado pela figura 3.

Figura 3 – Representação do conceito de duas camadas de pontos de referência.



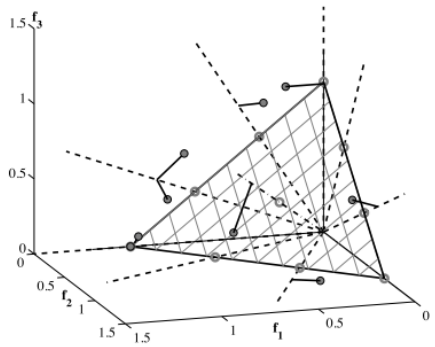

Fonte: (DEB; JAIN, 2014)

De acordo com Ribeiro (2018), o algoritmo NSGA-III possui como característica a priorização da diversidade de soluções uma vez que os membros da população estão associados

aos pontos de referência, onde estes estão amplamente distribuídos em todo hiperplano normalizado, fazendo assim com que as soluções obtidas também estejam distribuídas de forma ampla na fronteira de Pareto.

Ainda de acordo com Ribeiro (2018), a associação de cada solução à um ponto de referência é feita através de uma reta de referência para cada ponto do hiperplano partindo-se da origem e representada pela figura 4. Cada membro da população é associado ao ponto de referência que possuir a menor distância perpendicular entre eles. O processo de seleção das soluções se inicia após todas as soluções estarem associadas à um ponto de referência.

Figura 4 – Associação dos membros da população com os pontos de referência




Fonte: (DEB; JAIN, 2014)

### 3.6.1 O Algoritmo

O NSGA-III se inicia com a criação de uma população aleatória de tamanho N e um conjunto de pontos de referência m dimensionais distribuídos em um hiperplano. Espera-se que em cada ponto de referência seja encontrado um membro da população no decorrer das gerações, isto implica que as soluções estão diversificadas e espaçadas no hiperplano.

O loop principal é então executado, gerando filhos através de cruzamento e realizando a mutação. Em seguida, é feita a seleção da próxima geração escolhendo os filhos que mais se aproximam das retas de referência (RIBEIRO, 2016b).

O processo de seleção dos melhores membros para a próxima geração classifica a população de pais e filhos em níveis de não-dominância, também conhecidos como fronteiras, onde a fronteira $F1$ é composta pelos melhores indivíduos que não foram dominados por nenhum outro e que melhor aproximam da fronteira de Pareto. A próxima geração é composta por $n$ melhores indivíduos ordenados pelas fronteiras iniciando em $F1$, sendo $n$ o tamanho limite da população (DEB; JAIN, 2014).

**Algorithm 1** NSGA-III - PROCEDIMENTO DA GERAÇÃO $t$

**Entrada:** Pontos de referência $H$, População de pais $P_t$
População $P_{t+1}$
**inicio**
$S_t = \emptyset, i = 1$
$Q_t = recombinação + mutação(P_t)$
$R_t = P_t \cup Q_t$
$(F_1, F_2, ...) =$ non_dominated_sort$(R_t)$

$|S_t| \geq N$ $S_t = S_t \cup F_i$
$i = i + 1$ Última fronteira a ser incluída: $F_l = F_i$
$|S_t| = N$ $P_{t+1} = S_t$
$P_{t+1} = \cup_{j=1}^{l-1} F_j$
Pontos a serem escolhidos de $F_l : K = N - |P_{t+1}|$
Normalize os objetivos e crie o conjunto de referência $Z^r : Normalize(f^n, S_t, Z^r, Z^s, Z^a)$
Associe cada membro de $S_t$ a um ponto de referência: $[\pi(s), d(s] = Associação(S_t, Z^r)$ onde; $\pi(s)$: ponto de referência mais próximo, $d$: distancia entre $s$ e $\pi(s)$
Calcule a contagem de nicho do ponto de referência $j \in Z^r : p_j = \sum s \in S_t/F_l((\pi(s) = j)?1 : 0)$
Escolha $K$ membros um por vez de $F_l$ para construir $P_{t+1} : Niching(K, p_j, \pi, d, Z^r, F_l, P_{t+1}$
**fin**

## 3.7 Funções de Teste

De acordo com (DEB et al., 2001), algoritmos evolucionários de muitos objetivos têm como responsabilidade alcançar a convergência em direção à fronteira de Pareto e encontrar uma distribuição de soluções ótimas e diversificadas em toda a Pareto.

As funções ou problemas de testes servem para desafiar os algoritmos sob essas duas responsabilidades, criando obstáculos artificiais que dificultam sua convergência para Pareto. Uma das formas de criar esses obstáculos é através das características desejáveis analisadas por Huband et al. (2006), algumas delas são citadas abaixo:

1. Não utilizar parâmetros extremos;

   Parâmetros extremos são facilmente otimizados "por acidente"por exemplo quando os algoritmos evolutivos empregam estratégias de mutação truncando valores de parâmetros de volta para a borda de seu domínio.

2. Não utilizar parâmetros medianos;

   Algoritmos Evolutivos que empregam uma população inicial de vetores de parâmetros inicializados uniformemente aleatoriamente e que empregam recombinação intermediária, podem ser inclinados a encontrar soluções ótimas caso o problema inclua parâmetros medianos.

3. Número de parâmetros escalonáveis;

   Permitir a escalabilidade do número de parâmetros é benéfico para testar os diferentes níveis de dificuldade.

4. Número de objetivos escalonáveis;

   O problema de teste deve ser escalonável de forma a ter N objetivos. O número de objetivos influencia diretamente a dificuldade do problema de teste. Dessa forma, ao se criar suítes de funções de teste, é desejável permitir que número de objetivos seja variável.

5. Conhecer a geometria da Pareto Ótima;

   A geometria da Pareto ótima pode ser linear, convexa, côncava, dentre outras combinações e pode influenciar diretamente na performance dos Algoritmos Evolutivos.

6. Conhecer a fronteira de Pareto.

   Muitas métricas de performance requerem conhecimento prévio da fronteira de Pareto. Conhecer a localização da fronteira se torna necessário para se avaliar com precisão o desempenho geral de qualquer algoritmo utilizado.

Além das características citadas acima, Deb et al. (2001) diz que fazem parte desses obstáculos a criação da fronteira de Pareto não convexa e discreta, e manter uma densidade variada de soluções ao longo da fronteira.

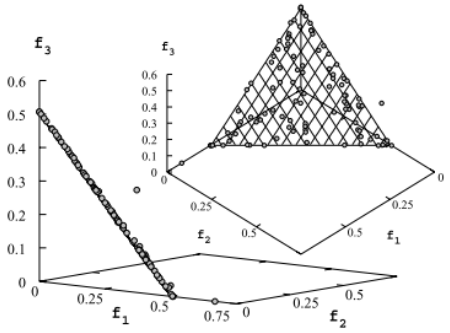
### 3.7.1 Função DTLZ1

A função de teste DTLZ1 proposta por Deb et al. (2001) faz parte de um conjunto de funções DTLZ cujo objetivo dos autores foi desafiar os algoritmos multiobjetivos (MOEA's) a mostrarem a eficácia em lidar com funções escalonáveis, tendo mais de dois objetivos. Os MOEA's demonstraram a capacidade em lidar com problemas de até dois objetivos de forma eficiente, portanto surgiu a necessidade de investigar o comportamento desses algoritmos ao lidarem com as funções escalonáveis. (DEB et al., 2001)

O problema DTLZ1 possui escalabilidade para $M$ objetivos e uma pareto ótima linear. A equação matemática abaixo representa esse problema:

$$DTLZ1 = \begin{cases} \text{Min } f_1(X) = \frac{1}{2}x_1x_2\cdots x_{M-1}(1+g(x_M)), \\ \text{Min } f_2(X) = \frac{1}{2}x_1x_2\cdots(1-x_{M-1})(1+g(x_M), \\ \vdots \\ \text{Min } f_{M-1}(X) = \frac{1}{2}x_1(1-x_2)(1+g(x_M)) \\ \text{Min } f_M(X) = \frac{1}{2}(1-x_1)(1+g(x_M)), \\ \text{sujeito a } 0 \leq x_i \leq 1, \text{ para } i = 1, 2, ..., n. \end{cases} \tag{3.4}$$

Figura 5 – Demonstração da Pareto da função DTLZ1 e a aproximação de pontos não dominados utilizando o NSGA-II ao otimizar a função




Fonte: (DEB et al., 2001)

## 3.8 *Inverted Generational Distance* (IGD)

O Inverted Generational Distance (IGD), tem sido considerado como um indicador de desempenho confiável para quantificar a convergência e a diversidade de algoritmos evolutivos de múltiplos e muitos objetivos. Utiliza-se este indicador em cada geração para selecionar as soluções com convergência e diversidade favoráveis (SUN; YEN; YI, 2018).

O IGD mede a distância total entre as soluções encontradas e as soluções ótimas da Fronteira de Pareto, sendo capaz de analisar a diversidade das soluções de forma a realizar uma avaliação aprimorada do algoritmo. Se o indicador possuir um valor muito baixo, isso significa que a solução esta bem próxima da ótima, servindo como métrica de avaliação do algoritmo em resolver um problema específico (ZHANG et al., 2008) (RIBEIRO, 2016a).

A fórmula do IGD é definida abaixo:

$$IGD = (F_a, F_r) = \frac{1}{\parallel F_r \parallel} \sum_{f_r \epsilon F_r} dist(f_r, F_a) \tag{3.5}$$

# 4 Desenvolvimento

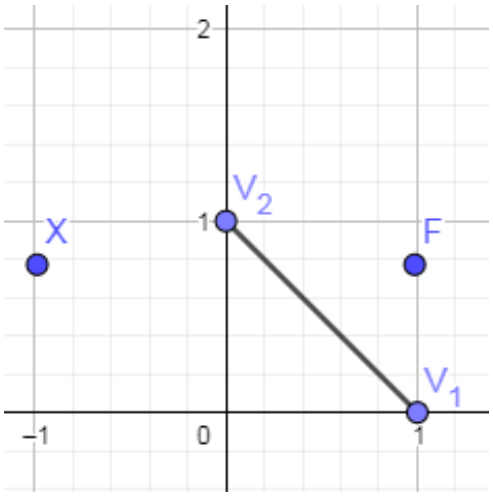
## 4.1 Especificação da função DNO$_f$

Este capítulo tem como objetivo explicar os conceitos que moldaram a criação da função de teste DNO$_f$, idealizada e descrita aqui pelo orientador deste trabalho. A função de teste é essencial para que o algoritmo evolucionário seja avaliado, criando cenários de alta complexidade, dificultando assim a convergência e a diversidade das soluções encontradas.

A função proposta considera uma entrada $x$ qualquer no espaço $R^M$, e sua saída também no espaço $R^M$, logo, $f : R^M \rightarrow R^M$. De forma simples, esta função transporta a entrada $x$ para o primeiro quadrante do sistema, obtendo as saídas $F = \{f_1, f_2, ..., f_n\}$. Cada $f_i$ é uma função objetiva a ser otimizada. Sua representação matemática é dada, de forma geral, pela equação 4.1 e graficamente podemos perceber seu uso nas figuras 6 e 7 sobre um $x$ qualquer no espaço $R^2$ e $R^3$ respectivamente.

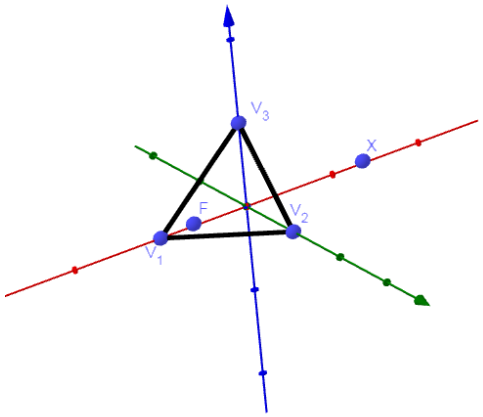
$$f_i = |x_i| \tag{4.1}$$

Figura 6 – Demonstração da função proposta em 2D




Fonte: A autora

Figura 7 – Demonstração da função proposta em 3D




Fonte: A autora

Ao analisar as figuras anteriores, percebe-se que há uma sobreposição do espaço das variáveis e dos objetivos, onde o espaço dos objetivos representa em seus eixos as funções objetivo do problema. Para introduzir a superfície das soluções não dominadas será utilizada a equação 4.2 de um plano ou hiperplano se a equação for generalizada para o $R^M$. Este hiperplano pode ser visualizado se ele for desenhado contendo os pontos $V_1$ e $V_2$ caso a saída seja apenas dois objetivos ou $V_1$, $V_2$ e $V_3$ caso seja três objetivos, as figuras 6 e 7 ilustram respectivamente estes casos.

$$\pi : f_1 + f_2 + ... + f_n - 1 = 0 \tag{4.2}$$

A função DNO$_f$ não permite que pontos $F$ estejam abaixo do hiperplano definido acima. Com este propósito, pontos que se encontram nesta localidade devem ser espelhados utilizando o próprio hiperplano. Para esta tarefa alguns passos serão realizados: encontrar a distância $d$ entre o ponto $F$ e o hiperplano $\pi$; e utilizar a direção do vetor normal $\overrightarrow{v}$ perpendicular a este hiperplano para espelhar $F$. O vetor $\overrightarrow{v}$ deve ser redimensionado para o comprimento igual a $2d$.

### 4.1.1 Distância entre $F$ e $\pi$

De acordo com DQ (2018), a equação 4.5 define o cálculo geral da distância entre um ponto $P_0$ e um plano $\pi'$, ambos definidos no $R^3$ e respectivamente representados pelas equações 4.3 e 4.4.

$$P_0 = (x_0, y_0, z_0) \tag{4.3}$$

$$\pi' : ax + bx + cz + d = 0 \tag{4.4}$$

$$d(P_0, \pi') = \frac{|a * x_0 + b * x_1 + c * x_2 + d|}{\sqrt{a^2 + b^2 + c^2}} \tag{4.5}$$

Adaptando a equação geral da distância entre um ponto e um plano para a necessidade aqui apresentada, onde o espaço é definido no $R^M$, $F$ é o ponto, o hiperplano é $\pi$ e $\overrightarrow{v} = (1, 1, 1, ...)$, a equação 4.6 é obtida. Note que o vetor $\overrightarrow{v}$ é formado por uma quantidade $M$ de números 1.

$$d(F, \pi) = \frac{|f_0 + f_1 + ... + f_n - 1|}{\sqrt{1^2 + 1^2 + 1^2}} \tag{4.6}$$

A equação 4.8 pode ser obtida simplificando a 4.6 se o valor de $s$, definido pela equação 4.7, for considerado.

$$s = \sum_{i=1}^{M} f_i \tag{4.7}$$

$$d(F, \pi) = \frac{|s - 1|}{\sqrt{M}} \tag{4.8}$$

### 4.1.2 Redimensão de $\overrightarrow{v}$

Sendo $D$ o comprimento do vetor $\overrightarrow{v}$ e $t$ é o valor escalar que multiplicado por $\overrightarrow{v}$ resultará em um vetor $\overrightarrow{v}'$ cujo o comprimento é $2d$ temos as equações 4.9, 4.10 e 4.11:

$$t * D = 2 * d \tag{4.9}$$

$$t * \sqrt{M} = 2 * \frac{|s - 1|}{\sqrt{M}} \tag{4.10}$$

$$t = 2 * \frac{|s - 1|}{M} \tag{4.11}$$

$$F' = F + 2 * \frac{|s - 1|}{M} * \overrightarrow{v} \tag{4.12}$$

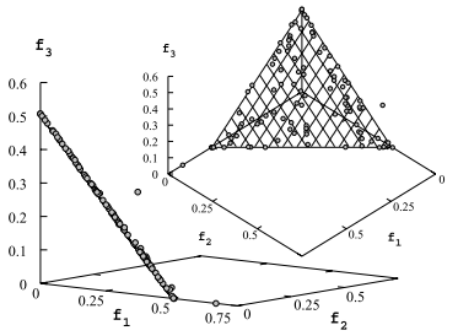
### 4.1.3 A função

O valor de $s$, definido na equação 4.7 pode ser usado para verificar a posição do ponto $F$ em relação ao hiperplano $\pi$: caso $s$ seja igual a uma unidade, $F$ pertence ao conjunto das soluções não dominadas, ou seja, $F$ se encontra exatamente no plano; caso seja maior que uma unidade, $F$ está acima do plano e é caracterizada por ser uma solução dominada; caso contrário, é um ponto que necessita ser espelhado. Desta forma a função proposta é representada pela equação 4.13.

$$F_i = \begin{Bmatrix} |X_i|, s >= 1, & \text{else} \\ |X_i| + \frac{(2 * |s-1|)}{M} & \end{Bmatrix} \tag{4.13}$$

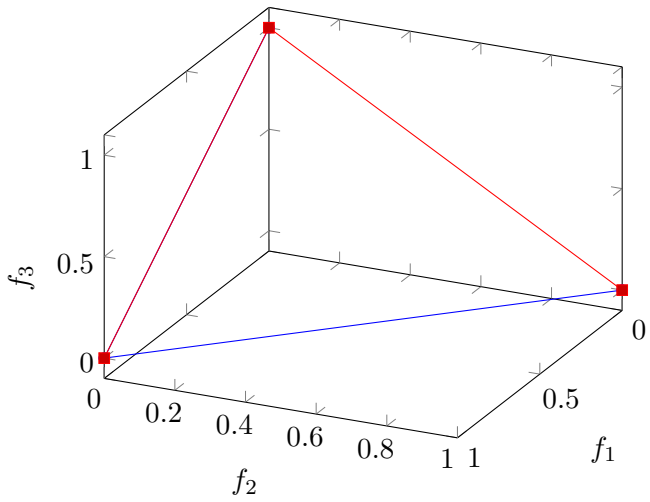
Existe uma similaridade entre a Pareto da função proposta e a Pareto da função DTLZ1 proposta por Deb et al. (2001) conforme demonstra as Figuras 8 e 9. A Pareto da função DTLZ1 possui seus pontos extremos fixos em 0.5, enquanto a pareto da função proposta possui seus pontos extremos fixos em 1. Devido a similaridade na geometria das fronteiras de Pareto das funções citadas acima, foi adotado para fins de comparação a otimização do algoritmo NSGA-III com a função DTLZ1.

Figura 8 – Demonstração da Fronteira de Pareto da função de teste DTLZ1




Fonte: (DEB et al., 2001)

Figura 9 – Pareto da Função $DNO_f$




Fonte: A autora

## 4.2 Parâmetros de Execução do NSGA-III

A execução dos testes com as funções DNO$_f$ e DTLZ1, foi baseada na parametrização dos testes realizados por Deb e Jain (2014) ao realizarem uma sequência de execuções do algoritmo NSGA-III para otimizar várias funções de muitos objetivos em seguida retornando os valores piores, melhores e medianos de acordo com o cálculo do IGD, exibindo os resultados em uma tabela comparativa.

Para obtenção de resultados, foi realizada a parametrização do algoritmo NSGA-III composto no *framework* JMetal e a implementação da função $\text{DNO}_f$. As duas funções foram executadas com 3, 5, 8, 10 e 15 objetivos. Para cada objetivo, foi necessário executar um número específico de gerações conforme demonstra a tabela 1.

Tabela 1 – Número de gerações por objetivo

| **Objetivos** | **Nº de Gerações** |
|---|---|
| 3 | 400 |
| 5 | 600 |
| 8 | 750 |
| 10 | 1000 |
| 15 | 1500 |

Cada conjunto de gerações foi executado 20 vezes para cada objetivo da tabela acima, com as duas funções. O objetivo principal de se escolher 20 execuções para cada objetivo foi para se igualar aos testes realizados por Deb e Jain (2014) ao realizar o mesmo procedimento para testar o algoritmo NSGA-III proposto.

Os pontos de referência foram calculados conforme a fórmula abaixo, onde $A = m + p - 1$ e $p$ é a quantidade de subdivisões por eixo e $m$ a quantidade de objetivos.

$$H = \left( \frac{A!}{p!(A-p)!} \right) \tag{4.14}$$

A Tabela 2 demonstra a configuração dos parâmetros do NSGA-III conforme o acréscimo de objetivos. É possível perceber que à medida que o número de objetivos aumenta, o numero de gerações também aumenta, tornando o processo de otimização mais complexo.

Tabela 2 – Configuração dos Parâmetros do NSGA-III

| **Qtd. de Objetivos (M)** | **Tam. da População (N)** | **Qtd. de Pontos de Referência (H)** | **Qtd. de Subdivisões por Eixo (p)** |
|---|---|---|---|
| 3 | 92 | 91 | 12 |
| 5 | 212 | 210 | 6 |
| 8 | 156 | 156 | p1 = 3<br>p2 = 2 |
| 10 | 276 | 275 | p1 = 3<br>p2 = 2 |
| 15 | 136 | 135 | p1 = 2<br>p2 = 1 |

Foram realizados diversos testes alterando o número de variáveis em relação ao número de objetivos, conforme demonstrado na Tabela 3. Cada configuração representada na Tabela 3 foi executada 20 vezes. A escolha da quantidade de execuções a se fazer foi baseada nos testes realizados por Deb e Jain (2014). Dessa forma, é possível obter uma média dos resultados obtidos e evitar que haja algum resultado fora do padrão, obtendo mais assertivamente os valores ótimos.

Tabela 3 – Parametrização de variáveis e objetivos

| **Qtd de objetivos** | **Teste 1 - $DNJ_F$ (Variáveis)** | **Teste 2 - $DNJ_F$ (Variáveis)** | **Teste 1: DTLZ1 (Variáveis)** | **Teste 2: DTLZ1 (Variáveis)** |
|---|---|---|---|---|
| 3 | 3 | 7 | 3 | 7 |
| 5 | 5 | 9 | 5 | 9 |
| 8 | 8 | 12 | 8 | 12 |
| 10 | 10 | 14 | 10 | 14 |
| 15 | 15 | 19 | 15 | 19 |

### 4.2.1 Cálculo da Pareto

O algoritmo utilizado para calcular a Pareto da função DNO$_f$ foi proposto por Chiang (2014). O autor também propôs uma representação do algoritmo NSGA-III na linguagem C++, que não foi utilizado neste trabalho.

Foi realizado uma adaptação do algoritmo que gera a Pareto ótima da suíte DTLZ para gerar também a Pareto da DNO$f$. Após obter os arquivos contendo a Pareto da função proposta, os mesmos foram utilizados no framework JMetal para realizar a otimização através do NSGA-III.

# 5 Resultados Obtidos

Esta seção apresenta os principais resultados obtidos durante o desenvolvimento deste trabalho, demonstrando mais especificamente o resultado da otimização da função DNO$_f$ e DTLZ1 em cenários de 3, 5, 8, 10 e 15 objetivos. Dados comparativos foram gerados e são demonstrados na tabela 4.

Os testes foram realizados no sistema operacional Windows 7 Professional - 64 bits, 8GB RAM, processador Intel Core I5-3230M. Para execução e customização do JMetal foi utilizada a IDE Intellij IDEA Community Edition 2017.3.4. É possível encontrar o framework JMetal através do site: https://jmetal.github.io/jMetal/.

Foram gerados vinte conjuntos de soluções para o testes 1 e 2 das funções DNO$_f$ e DTLZ1, conforme descrição na tabela 3. Cada execução implementa um numero pré definido de gerações conforme demonstrado na tabela 2, e retorna o valor final do IGD para aquele conjunto de gerações.

Ao final de cada teste, foi gerado um arquivo contendo os valores do IGD para cada uma das vinte execuções e em seguida foi calculado o IGD mínimo, médio e máximo de cada objetivo. Para montar a tabela comparativa, foram utilizados os IGD's mínimo, médio e máximo referentes a cada variação de objetivos e variáveis, conforme demonstra a tabela 4.

O IGD é uma métrica que a avalia a capacidade que um algoritmo de otimização multiobjetivo tem para resolver problemas complexos e achar valores próximos da Pareto ótima, portanto, quanto menor for o IGD, menor será a distância entre a solução ótima e a solução encontrada, significando que o algoritmo conseguiu valores ótimos com boa convergência.

A tabela 4 demonstra que, para cada configuração de objetivos e variáveis, a função DNO$_f$ apresentou o maior valor para o IGD, e consequentemente trouxe ao NSGA-III uma dificuldade maior ao realizar a otimização.

O aumento do número de variáveis em relação ao número de objetivos trouxe como resultados os IGD's médios maiores em ambas as funções e em todos os cenários. Destaca-se a variação máxima de IGD's obtida pela função DNO$_f$ no cenário de 15 objetivos, obtendo uma diferença de 0.2668 entre os IGD's mínimo e máximo no para 19 variáveis em relação à DTLZ1 que apresentou a diferença de 0.008 no mesmo caso. Isso demonstra que o NSGA-III teve mais consistência na convergência para a DTLZ1 e maior dificuldade de se estabilizar em um valor de IGD médio ao otimizar a função DNO$_f$ proposta com alto número de objetivos e variáveis.

Tabela 4 – IGD's mínimos, médios e máximos obtidos pelo NSGA-III

| **Objetivos** | **Max Gerações** | **Variáveis** | **DNO$_f$** | **DTLZ1** |
|---|---|---|---|---|
| 3 | 400 | 3 | 1.946E-01 | 2.026E-02 |
| | | | 2.038E-01 | 2.037E-02 |
| | | | 2.161E-01 | 2.098E-02 |
| | | 7 | 1.855E-01 | 2.028E-02 |
| | | | 2.120E-01 | 2.089E-02 |
| | | | 2.399E-01 | 2.414E-02 |
| 5 | 600 | 5 | 2.701E-01 | 2.363E-04 |
| | | | 2.873E-01 | 1.104E-03 |
| | | | 3.030E-01 | 4.255E-03 |
| | | 9 | 2.770E-01 | 7.097E-04 |
| | | | 2.939E-01 | 5.477E-03 |
| | | | 3.123E-01 | 3.507E-02 |
| 8 | 750 | 8 | 3.293E-01 | 9.947E-04 |
| | | | 3.549E-01 | 1.396E-03 |
| | | | 3.931E-01 | 2.562E-03 |
| | | 12 | 3.412E-01 | 2.636E-03 |
| | | | 3.998E-01 | 5.417E-03 |
| | | | 4.448E-01 | 7.965E-03 |
| 10 | 1000 | 10 | 3.295E-01 | 1.240E-03 |
| | | | 3.597E-01 | 3.304E-03 |
| | | | 3.864E-01 | 3.240E-02 |
| | | 14 | 3.508E-01 | 2.715E-03 |
| | | | 4.001E-01 | 4.367E-03 |
| | | | 4.636E-01 | 8.759E-03 |
| 15 | 1500 | 15 | 3.703E-01 | 1.356E-03 |
| | | | 4.690E-01 | 1.715E-03 |
| | | | 5.420E-01 | 2.399E-03 |
| | | 19 | 3.352E-01 | 3.554E-03 |
| | | | 4.937E-01 | 6.301E-03 |
| | | | 6.020E-01 | 1.159E-02 |

## 5.1 IGD's Obtidos

Nessa seção, serão demonstrados os gráficos contendo os IGD's mínimo, médio e máximo para as duas funções testadas em cada objetivo.

A visualização dos gráficos permite compreender as diferenças entre os IGD's das duas funções em relação ao número de variáveis escolhidas, e, a partir daí, obter conclusões a respeito do nível de dificuldade de cada configuração e função testada.

Figura 10 – Variação do IGD - 3 objetivos

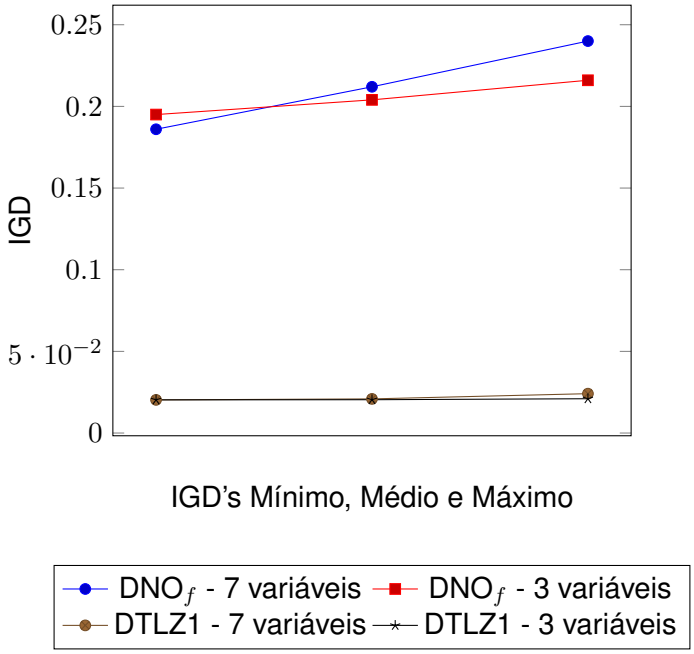



Figura 11 – Variação do IGD - 5 objetivos

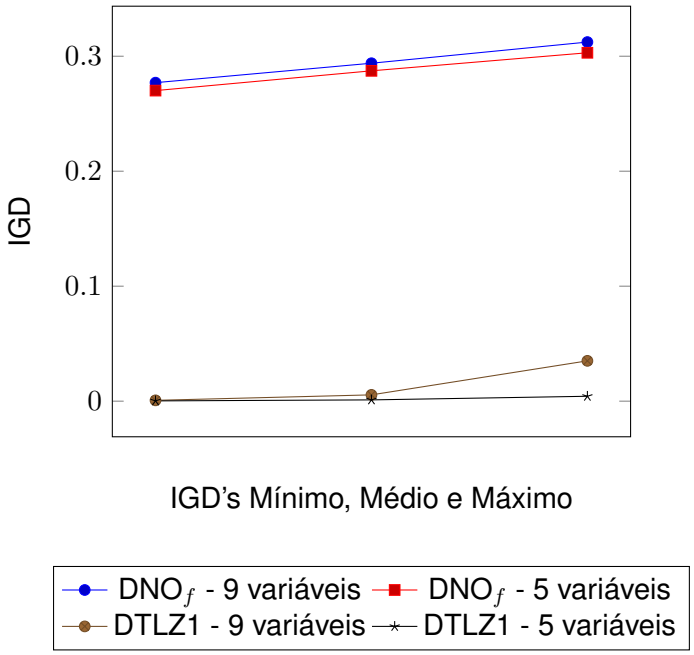


Figura 12 – Variação do IGD - 8 objetivos

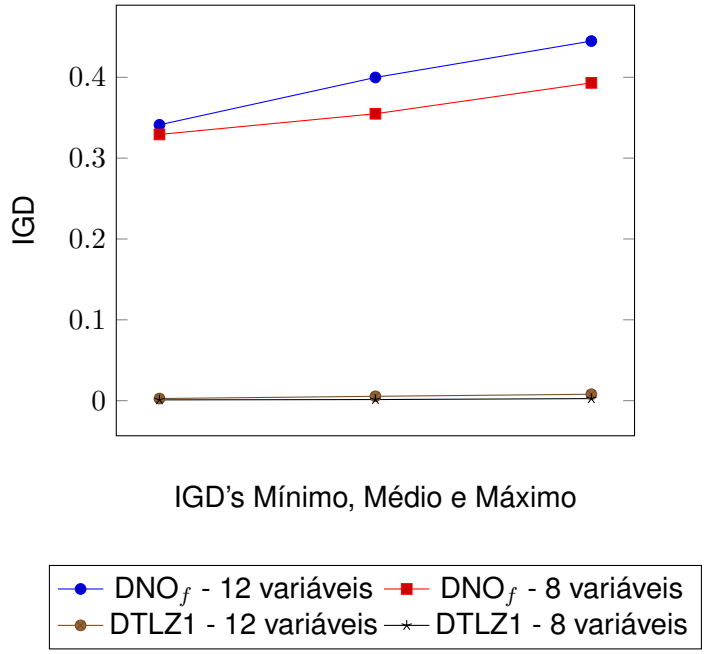



Figura 13 – Variação do IGD - 10 objetivos

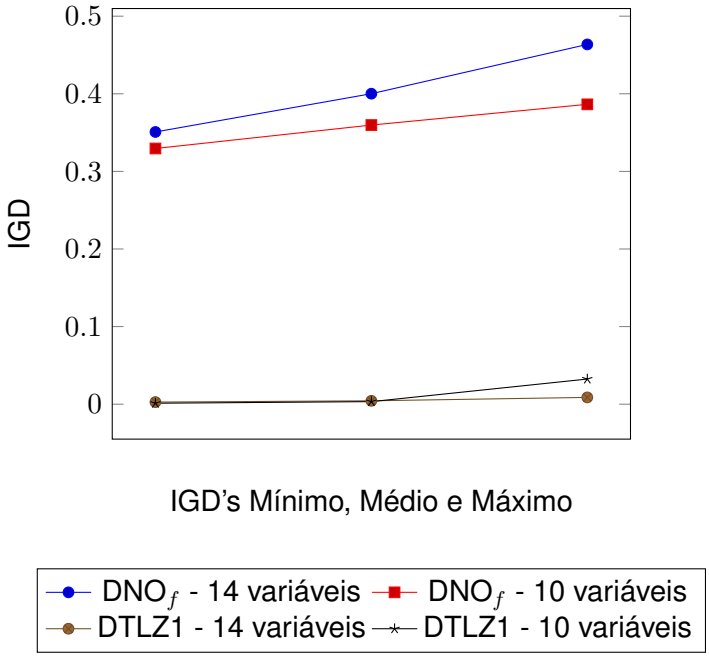
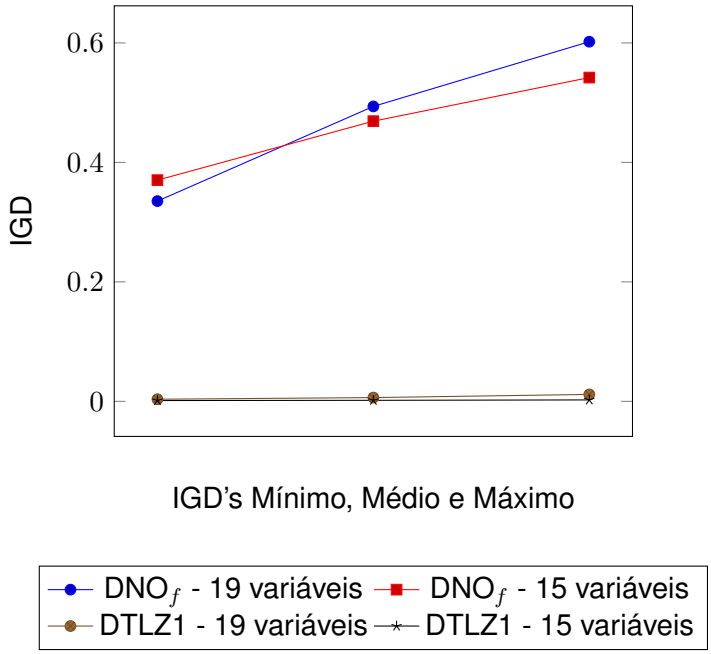


Figura 14 – Variação do IGD - 15 objetivos




## 5.2 Soluções Obtidas - Aproximação da Pareto

As imagens abaixo representam as soluções ótimas obtidas ao final da última geração, ao otimizar as funções DNO$_f$ e DTLZ1 para três objetivos. Não é possível representar os demais objetivos em um espaço tridimensional, portanto esta seção ficou limitada apenas a três objetivos.

Os gráficos apresentam pontos de referência igualmente distribuídos. As melhores soluções são as que se encontram espalhadas e próximas de cada ponto.

No cenário apresentado na figura 15, percebe-se uma instabilidade de soluções ótimas, algumas muito próximas umas das outras e outras muito distantes da Pareto, com um espalhamento mediano. Em contrapartida, as soluções da figura 16 apresentam um bom espalhamento e soluções mais próximas dos pontos de referência. Uma explicação para essa discrepância de resultados pode ser dada pelo aumento do número de variáveis na primeira figura, o que a tornou mais complexa no cenário de otimização e desestabilizou os resultados.

Os gráficos 17 e 18 referentes a função DTLZ1, apresenta soluções ótimas perfeitamente localizadas em cada um dos pontos de referência da Pareto, isso demonstra que a DTLZ1 foi uma função que agregou baixa complexidade para o algoritmo de otimização NSGA-III, ou seja, não gerou instabilidades ou dificuldades durante o processo.

Figura 15 – Aproximação da função DNO$_f$ - 3 objetivos, 7 variáveis

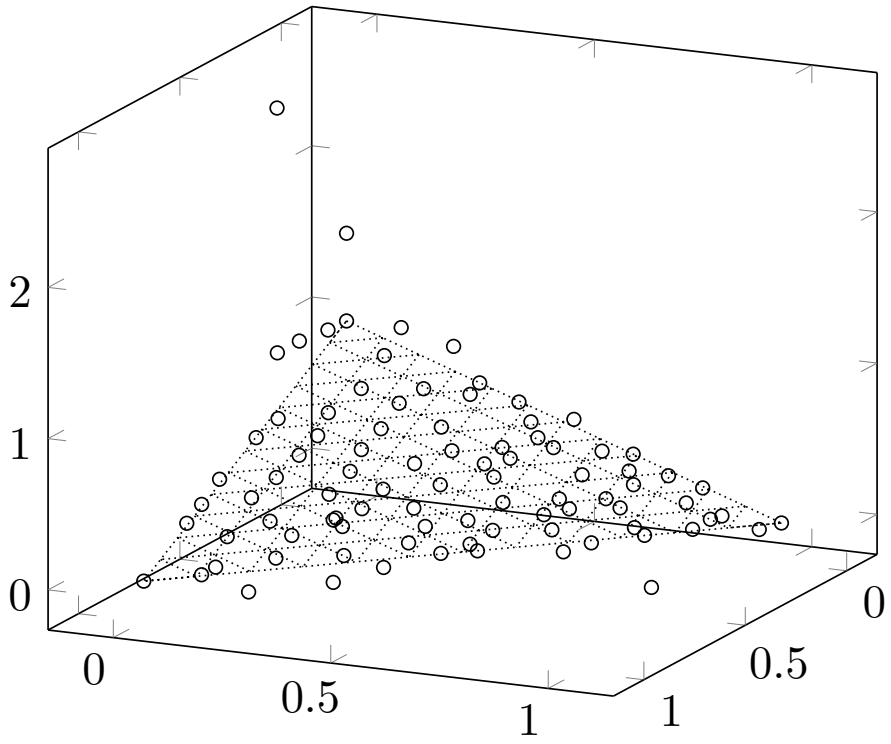



Figura 16 – Aproximação da função DNO$_f$ - 3 objetivos, 3 variáveis

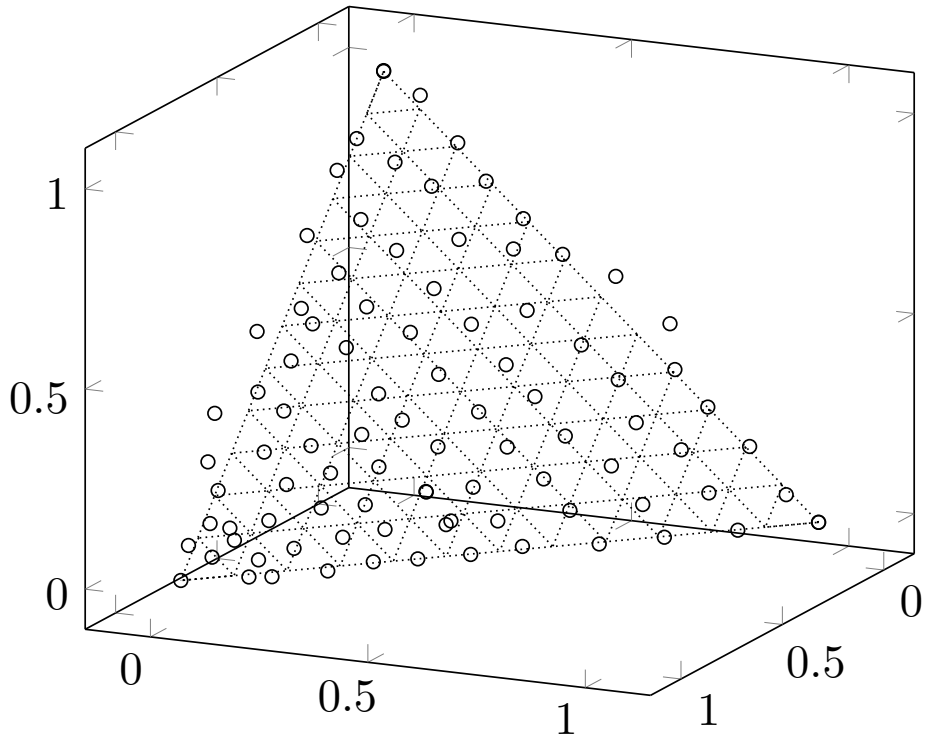


Figura 17 – Aproximação da função DTLZ1 - 3 objetivos, 7 variáveis

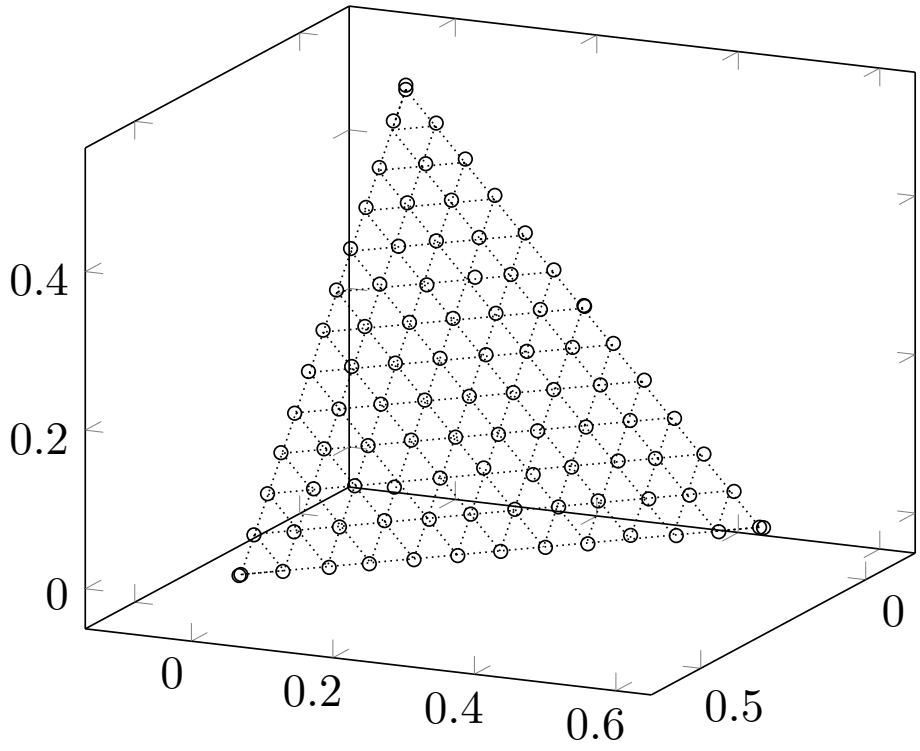
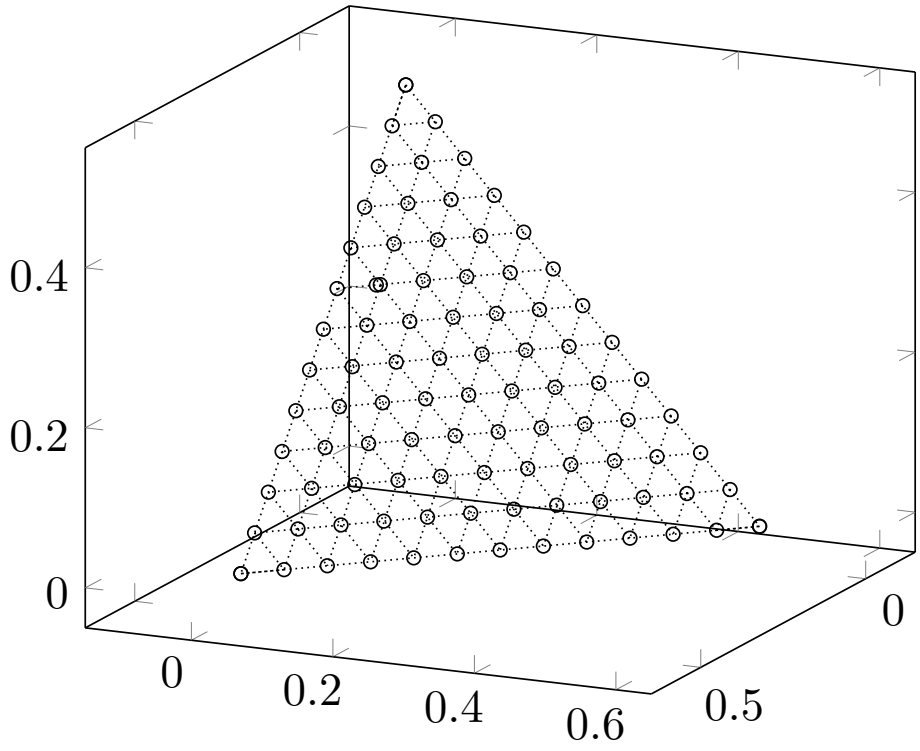



Figura 18 – Aproximação da função DTLZ1 - 3 objetivos, 3 variáveis



# 6 Conclusão

A partir dos resultados obtidos, percebe-se que a função proposta neste trabalho apresentou maior nível de dificuldade ao algoritmo NSGA-III ao tratar problemas com até quinze objetivos.

Quando comparada à função DTLZ1, que já é referência no campo da computação evolucionária, houve diferenças mais discrepantes a partir do cenário de oito objetivos, onde a variação entre os IGD's mínimo e máximo aumentou significativamente em especial quando as variáveis foram maiores que o número de objetivos.

Conclui-se com este trabalho que a criação de novas funções teste podem agregar valor em um cenário de muitos objetivos, principalmente quando se deseja aumentar a complexidade dos testes de um algoritmo e classificá-lo quanto a convergência para diferentes tipos de funções. Uma função teste sempre trará diferentes níveis de dificuldade a um algoritmo, mas nem sempre será possível realizar uma comparação justa com outras funções já existentes, pois existem diversas configurações que diferenciam umas das outras, como por exemplo a geometria da Pareto, a dimensionalidade, o número de variáveis de entrada, dentre outros. Um algoritmo que se adapta em resolver diferentes funções teste, está mais propenso a ser um algoritmo bem avaliado e bem utilizado no mundo real, pois os problemas atuais estão cada vez mais difíceis de serem tratados. Moldar novas funções também significa acrescentar ao mundo acadêmico novos cenários de teste, de forma a obter algoritmos melhor preparados para lidar com problemas de diferentes escalas e áreas de atuação, como por exemplo na indústria, robótica, biologia, dentre outros.

A função proposta contribui para o cenário de avaliação de algoritmos de muitos objetivos, podendo ser utilizada em outros algoritmos além do NSGA-III. A função pode apresentar diferentes níveis de dificuldade para diferentes algoritmos, diversificando os resultados. O comportamento de cada algoritmo será diferente, podendo fazer com que a função seja mais complexa para uns e menos complexa para outros.

Por ser uma função de fácil customização, fica a critério do usuário modificá-la de forma a simular características como a multimodalidade ou n-dimensionalidade e observar quais são os algoritmos que melhor otimizam cada tipo de customização.

## 6.1 Trabalhos Futuros

Sugere-se como continuação deste trabalho:

1. Utilizar novos algoritmos de múltiplos objetivos para otimizar a função DNO$_f$;
2. Investigar a complexidade de uma customização da função DNO$_f$, de forma a obter diferentes resultados com diferentes algoritmos;
3. Comparar a performance da função proposta com diferentes funções teste;

# Referências


BARBOZA, E. de A. Uma nova abordagem de otimização multiobjetivos para projeto de amplificador raman. 2011. Citado na página 15.

BRADSTREET, L. et al. Use of the wfg toolkit and pisa for comparison of moeas. In: IEEE. *Computational Intelligence in Multicriteria Decision Making, IEEE Symposium on*. [S.l.], 2007. p. 382–389. Citado na página 14.

BUI, L. T.; ALAM, S. An introduction to multi-objective optimization. *Multi-Objective Optimization in Computational Intelligence: Theory and Practice*, IGI Global Snippet, p. 1–19, 2008. Citado nas páginas 14, 15 e 18.

CHIANG, T.-C. *nsga3cpp: A C++ implementation of NSGA-III*. 2014. Disponível em: <https://web.ntnu.edu.tw/~tcchiang/publications/nsga3cpp/nsga3cpp.htm>. Citado na página 29.

CHO, H.-J. et al. A comparative study of teaching-learning-self-study algorithms on benchmark function optimization. *Korean Journal of Chemical Engineering*, Springer, v. 34, n. 3, p. 628–641, 2017. Citado na página 12.

COELLO, C. A. C.; LAMONT, G. B. *Applications of multi-objective evolutionary algorithms*. [S.l.]: World Scientific, 2004. v. 1. Citado na página 18.

COELLO, C. C. Evolutionary multi-objective optimization: a historical view of the field. *IEEE computational intelligence magazine*, IEEE, v. 1, n. 1, p. 28–36, 2006. Citado na página 16.

DAS, I.; DENNIS, J. E. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, SIAM, v. 8, n. 3, p. 631–657, 1998. Citado na página 19.

DEB, K. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary computation*, MIT Press, v. 7, n. 3, p. 205–230, 1999. Citado nas páginas 12 e 17.

DEB, K. *Multi-objective optimization using evolutionary algorithms*. [S.l.]: John Wiley & Sons, 2001. v. 16. Citado nas páginas 11 e 18.

DEB, K. Multi-objective optimization using evolutionary algorithms: an introduction. *Multi-objective evolutionary optimisation for product design and manufacturing*, p. 1–24, 2011. Citado na página 15.

DEB, K.; JAIN, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Trans. Evolutionary Computation*, v. 18, n. 4, p. 577–601, 2014. Citado nas páginas 11, 12, 17, 18, 19, 20, 27 e 28.

DEB, K. et al. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, IEEE, v. 6, n. 2, p. 182–197, 2002. Citado nas páginas 11 e 18.

DEB, K. et al. Scalable test problems for evolutionary multi-objective optimization. *TIK-Report*, Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK), v. 112, 2001. Citado nas páginas 11, 12, 14, 21, 22, 23 e 27.

DQ, N. *Distance from point to plane.* 2018. Disponível em: <https://mathinsight.org/distance_point_plane>. Citado na página 25.

EDGEWORTH, F. Y. Mathematical psychics (london: Kegan paul, 1881). *EdgeworthMathematical Psychics1881*, 1881. Citado na página 15.

FIGUEIREDO, E. M. d. N. Algoritmo baseado em enxame de partículas para otimização de problemas com muitos objetivos. Universidade Federal de Pernambuco, 2013. Citado nas páginas 16 e 17.

FLEMING, P. J.; PURSHOUSE, R. C.; LYGOE, R. J. Many-objective optimization: An engineering design perspective. In: SPRINGER. *International conference on evolutionary multi-criterion optimization*. [S.l.], 2005. p. 14–32. Citado na página 16.

HUBAND, S. et al. A scalable multi-objective test problem toolkit. In: SPRINGER. *International Conference on Evolutionary Multi-Criterion Optimization*. [S.l.], 2005. p. 280–295. Citado na página 14.

HUBAND, S. et al. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 10, n. 5, p. 477–506, 2006. Citado na página 21.

JAMIL, M.; YANG, X.-S. A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, Inderscience Publishers Ltd, v. 4, n. 2, p. 150–194, 2013. Citado nas páginas 12 e 14.

JUSTESEN, P. D. Multi-objective optimization using evolutionary algorithms. 2009. Citado nas páginas 11, 14 e 15.

LI, K. et al. An evolutionary many-objective optimization algorithm based on dominance and decomposition. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 19, n. 5, p. 694–716, 2015. Citado na página 11.

LOBATO, F. S. et al. Otimização multi-objetivo para o projeto de sistemas de engenharia. Universidade Federal de Uberlândia, 2008. Citado na página 11.

MACHADO, M. D. Algoritmo evolucionário pbil multi-objetivo aplicado ao problema da recarga de reatores nucleares. *DSc thesis, COPPE/UFRJ, Brazil*, 2005. Citado na página 18.

NAMETALA, C.; PAPPA, G. L.; CARRANO, E. Evolução diferencial multiobjetivo híbrido com k-means e nsga ii: Uma análise comparativa frente ao nsga iii. 10 2017. Citado na página 11.

NOCEDAL, J.; WRIGHT, S. Numerical optimization springer-verlag. *New York*, 1999. Citado na página 11.

OSYCZKA, A. Multicriteria optimization for engineering design. In: *Design optimization*. [S.l.]: Elsevier, 1985. p. 193–227. Citado na página 15.

PARETO, V. Cours d'économie politique, vol. i-ii, f. rouge, lausanne, trad. it. *Corso di economia politica*, 1896. Citado na página 15.

POLONI, C. Hybrid ga for multi objective aerodynamic shape optimisation. John Wiley & Sons Ltd, 1995. Citado na página 17.

RAO, S. S.; RAO, S. S. *Engineering optimization: theory and practice*. [S.l.]: John Wiley & Sons, 2009. Citado nas páginas 11 e 15.

RIBEIRO, M. G. Comparação de performance de variantes de mutação no differential evolution em problemas de otimização com muitos objetivos. 2018. Citado nas páginas 19 e 20.

RIBEIRO, R. d. S. *Comparação de Algoritmos Evolucionários para Problemas com Muitos Objetivos*. 2016. Citado na página 23.

RIBEIRO, R. de S. *Comparação De Algoritmos Evolucionários Para Problemas Com Muitos Objetivos*. 2016. Tese (Doutorado) — Universidade Federal de Minas Gerais, 2016. Citado nas páginas 19 e 20.

SAMPAIO, P. R. *Teoria, métodos e aplicações de otimização multiobjetivo*. 2011. Tese (Doutorado) — Universidade de São Paulo, 2011. Citado na página 14.

SCHEFFÉ, H. Experiments with mixtures. *Journal of the Royal Statistical Society. Series B (Methodological)*, JSTOR, p. 344–360, 1958. Citado na página 19.

SEKULSKI, Z. Multi-objective topology and size optimization of high-speed vehicle-passenger catamaran structure by genetic algorithm. *Marine Structures*, v. 23, n. 4, p. 405 – 433, 2010. ISSN 0951-8339. Disponível em: <http://www.sciencedirect.com/science/article/pii/S095183391000050X>. Citado nas páginas 15, 16 e 17.

SRINIVAS, N.; DEB, K. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, MIT Press, v. 2, n. 3, p. 221–248, 1994. Citado na página 11.

STORN, R.; PRICE, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, Springer, v. 11, n. 4, p. 341–359, 1997. Citado na página 11.

SUN, Y.; YEN, G. G.; YI, Z. Igd indicator-based evolutionary algorithm for many-objective optimization problems. *IEEE Transactions on Evolutionary Computation*, IEEE, 2018. Citado na página 23.

VESTERSTROM, J.; THOMSEN, R. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In: IEEE. *Evolutionary Computation, 2004. CEC2004. Congress on*. [S.l.], 2004. v. 2, p. 1980–1987. Citado na página 11.

WRIGHT, S.; NOCEDAL, J. Numerical optimization. *Springer Science*, v. 35, n. 67-68, p. 7, 1999. Citado na página 15.

YUAN, Y.; XU, H.; WANG, B. An improved nsga-iii procedure for evolutionary many-objective optimization. In: ACM. *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*. [S.l.], 2014. p. 661–668. Citado na página 18.

ZHANG, Q.; LI, H. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, IEEE, v. 11, n. 6, p. 712–731, 2007. Citado na página 11.

ZHANG, Q. et al. Multiobjective optimization test instances for the cec 2009 special session and competition. *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, v. 264, 2008. Citado na página 23.

ZITZLER, E. Evolutionary algorithms for multiobjective optimization: Methods and applications. Citeseer, 1999. Citado na página 17.

ZITZLER, E.; LAUMANNS, M.; THIELE, L. Spea2: Improving the strength pareto evolutionary algorithm. *TIK-report*, Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK), v. 103, 2001. Citado na página 11.