UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Otimização de Controladores *Fuzzy* por Algoritmos Genéticos Multiobjetivos no Domínio *Wavelet*

André Henrique Matias Pires

Orientador: Prof. Dr. Fábio Meneghetti Ugulino de Araújo

Tese de doutorado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e de Computação da UFRN (área de concentração: Automação e Sistemas) como parte dos requisitos para obtenção do título de Doutor em Ciências.

Natal, RN, 23 de outubro de 2023

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Central Zila Mamede

Pires, André Henrique Matias.
Otimização de controladores fuzzy por algoritmos genéticos multiobjetivos no domínio Wavelet / André Henrique Matias Pires. - 2023.
Orientação: Prof. Dr. Fábio Meneghetti Ugulino de Araújo.

Tese (doutorado) - Universidade Federal do Rio Grande do Norte, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica e de Computação. Natal, RN, 2023.
96 f.: il.

1. Controlador Fuzzy - Tese. 2. Wavelet - Tese. 3. Otimização de Controladores - Tese. 4. Otimização Multiobjetivos - Tese. 5. Algoritmo Genético de Ordenação Não Dominada II - NSGA-II - Tese. 6. Algoritmo Genético de Ordenação Não Dominada III - NSGA-III - Tese. I. Araújo, Fábio Meneghetti Ugulino de. II. Título.

RN/UF/BCZM CDU 004.312(043.2)

Elaborado por Fernanda de Medeiros Ferreira Aquino - CRB-15/301

Agradecimentos

A Deus, a Sua Virgem Mãe, a São José e a Santo Otávio. À minha esposa, Ana Carolina, que sempre me apoiou e incentivou, soube compreender a falta de tempo e me auxiliou nas correções do trabalho. À minha filha, Ana Cecília, que me deu forças para continuar na árdua batalha. Aos meus familiares em especial aos meus pais, Ilana e Adonizedeque e meus irmãos, Pedro e Miguel. Ao meu orientador, Fábio Meneghetti, e a todos os amigos do laboratório de controle de processos, em especial José Kleiton, Mário Sérgio, Willians e Leandro Augusto, sempre dispostos a auxiliar.

.

Resumo

Devido à crescente competitividade na indústria, torna-se imperativo o uso de técnicas de ajuste mais eficientes e que, de fato, possam encontrar controladores com o desempenho desejado. Com essa proposta, técnicas de otimização podem ser usadas para obter os parâmetros do controlador de acordo com um critério de avaliação, que deve codificar o quão bom é um determinado controlador, expressando adequadamente as especificações desejadas, para que o algoritmo empregado possa encontrar o controlador desejado. Os métodos tradicionalmente utilizados na sintonia apresentam uma dificuldade em expressar as especificações pretendidas. A dificuldade encontrada se deve a que os critérios tradicionalmente adotados, no geral, utilizam apenas a informação do erro total, através de índices como a Integral do Erro Absoluto (IEA) ou a Integral do Erro Quadrado (ISE), que não descrevem aspectos do comportamento do sistema, como se a resposta está muito agressiva e oscilatória, o erro de regime permanente, tempo de subida e tempo de estabilização, como faria um projetista humano. Além disso, algumas dessas impressões não estão bem definidas para referências diferente do degrau, carecendo de generalidade. Desse modo, o algoritmo de otimização responsável por obtenção dos parâmetros do controlador o faz de acordo com uma função de avaliação, a qual deve conseguir, de fato, codificar o quão bom é um dado controlador, expressando de forma adequada as especificações desejadas, de modo que o algoritmo de otimização empregado consiga encontrar o controlador que melhor satisfaça o problema apresentado. Em vista disso, será apresentada uma metodologia genérica de utilização da análise *wavelet* juntamente com técnicas de otimização multiobjetivo para se expressar o comportamento que se pretende alcançar pelo sistema controlado, de forma mais precisa e próxima da realizada pelo ser humano, permitindo uma otimização mais eficiente. Na metodologia proposta, a análise *wavelet*, muito presente na literatura, voltada para outras aplicações, sobretudo na análise de sinais, sons e imagens, é utilizada para obtenção de descritores que caracterizem aspectos do comportamento do sistema, como seu comportamento em regime permanente, comportamento no regime transitório, não amplificação de ruídos e rejeição a perturbações, esses descritores passam a constituir objetivos que serão otimizados por técnicas multiobjetivos. O estudo realizado utilizou técnicas de Algoritmo Genético Multiobjetivo (MOGAs) para a otimização, devido a serem amplamente utilizadas na literatura e por serem conhecidas por suas simplicidades e eficiências.

Palavras-chave: *Wavelet*, Otimização de Controladores, Otimização Multiobjetivos, NSGA-II, NSGA-III, Controlador *Fuzzy*.

Abstract

Due to the increasing competitiveness in the industry, it is imperative to use more efficient tuning techniques that can in fact find controllers with the desired performance. With this proposal, optimization techniques can be used to obtain the controller parameters according to an evaluation criterion, which should encode how good a particular controller is, properly expressing the desired specifications, so that the utilized algorithm can find the desired controller. The methods traditionally used in tuning present a difficulty in expressing the desired specifications. The difficulty is due to the fact that the traditionally used criteria, in general, only use the total error information, through indices such as the Integral Absolute Error (IAE) or the Integral Square Error (ISE), which do not describe aspects of system behavior, such as if the response is very aggressive and oscillatory, steady state error, rise time and stabilization time, as a human designer would do. Some of these impressions are not well defined for references other than the step, lacking generality. Thus the optimization algorithm responsible for obtaining the controller parameters according to an evaluation function, which must actually be able to encode how good a given controller is, adequately expressing the desired specifications, so that the optimization algorithm employed can find the controller that best satisfies such a function. In view of this, a generic methodology for using wavelet analysis will be presented along with multiobjective optimization techniques to express more closely and closely related to the human behavior of the controlled system, allowing a more accurate optimization. In the proposed methodology, wavelet analysis, very present in the literature, focused on other applications, especially in the analysis of signals, sounds and images, is used to obtain descriptors that describe aspects of system behavior, such as its steady state behavior, behavior In the transient, no amplification of noise and rejection of disturbances, these descriptors become objectives that will be optimized by multiobjective techniques. The study carried out used Multiobjective Genetic Algorithm (MOGAs) techniques for optimization, due to their being widely used in the literature and known for their simplicity and efficiency.

Keywords: Wavelet, Controller Optimization, Multi Objective Optimization, NSGA-II, NSGA-III, Fuzzy Controller.

Sumário

Lista de Figuras

Lista de Símbolos e Abreviaturas

ABC	Algoritmo de Colônia de Abelhas Artificiais
ACO	*Ant Colony Optimization*
CS	*Cuckoo Search*
DE	Evolução Diferencial
EA	Algoritmo Evolutivo
GA	*Genetic Algorithm*
GSA	*Gravitational Search Algorithm*
GSO	Genetic Swarm Optimization
GWO	*Gray Wolf Optimizer*
HLGA	Algoritmo Genético de Hajela e Lin
IEA	Integral do Erro Absoluto
IEAT	Integral do Erro Absoluto Ponderado no Tempo
ISDE	Immune self-adaptive differential evolution
ISE	Integral do Erro Quadrático
MOEA/D	Algoritmo Evolucionário Multiobjetivo Baseado em Decomposição
NPGA	Algoritmo genético Pareto nichado
NSGA	Algoritmo Genético de Ordenação Não Dominada
NSGA-II	Algoritmo Genético de Ordenação Não Dominada II
NSGA-III	Algoritmo Genético de Ordenação Não Dominada III
P	Controlador Proporcional
PD	Controlador Proporcional Derivativo
PI	Controlador Proporcional Integral

PID	Controlador Proporcional Integral Derivativo
PSO	*Particle Swarm Optimization*
SA	*Particle Swarm Optimization*
SPEA	O Algoritmo Evolutivo de Pareto Forte
STFT	*Short Time Fourier Transform*
TS	Tabu Search
VEGA	Algoritmo Genético Avaliado por Vetor

Capítulo 1

Introdução

Devido à crescente competitividade da indústria e à busca pelo aumento da produção, há uma progressiva necessidade do incremento da eficiência na fabricação dos mais diversos produtos. Tais incrementos possibilitam a redução dos custos de produção, minimizando, por sua vez, o custo final da mercadoria e reduzindo os insumos, com benefícios econômicos e para o meio ambiente. Uma estratégia que vem sendo usada nos últimos séculos para o aumento da eficiência da produção é o uso dos sistemas automáticos, que visam substituir a ação do homem na execução de diversas atividades, que, atualmente, são realizadas por computadores que executam tarefas de controle ou supervisão de processos industriais (ONOFRE, 2011).

Essa necessidade das fábricas melhorarem seus processos, produtos e serviços de modo a garantir a competitividade dos seus produtos e o aumento da produção (ONOFRE, 2011) gera a necessidade de otimização da sintonia dos controladores. É importante ressaltar que, mesmo uma pequena melhoria de desempenho pode resultar em aumentos consideráveis de lucro em algumas aplicações, é o caso da indústria de petróleo e gás, conforme apresentado em (CAMPOS; TEIXEIRA, 2006) em que, por exemplo, uma melhoria de $0,5\%$ no desempenho, em relação à recuperação de gás liquefeito de petróleo em uma unidade de processamento de gás natural, acarretou um aumento considerável nos ganhos anuais.

As técnicas de otimização são capazes de transportar o mecanismo de adaptação para construir um procedimento computacional para tratar de problemas de otimização de alta complexidade (CUNHA R. TAKAHASHI, 2012). Existe um grande número de técnicas de otimização desde as determinísticas, baseadas em valores numéricos exatos, como derivadas ou gradientes, ou aproximações destas, e técnicas heurísticas que são estocásticas, no sentido que em diferentes execuções um método retornará diferentes soluções subótimas. Porém, independentemente do algoritmo de otimização adotado para a otimização de controladores, é necessária a avaliação do seu desempenho e, no caso da sintonia automática, há uma necessidade de que essa avaliação de desempenho possa ser codificada matematicamente de modo a ser utilizada pelo algoritmo.

Nesse contexto, a utilização de índices como a integral do erro absoluto (IEA) e integral do erro absoluto ponderado no tempo (IEAT), são recorrentes na literatura (DOMAŃSKI *et al.*, 2020)(DORF *et al.*, 2005)(MARLIN, 1995). Essas funções de avaliação apresentadas na literatura funcionam de forma adequada quando o sinal de referência é um degrau. Quando

se deseja utilizar como sinal de referência uma rampa ou uma senoide, por exemplo, esse estudo se torna mais difícil ou mesmo impossível, como seria o caso do IEAT. Além disso, esses índices de avaliação, mesmo que usados em conjunto através de sua combinação linear, têm dificuldade de descrever o que é pretendido por um operador humano, que no geral, para um degrau, seria uma resposta pouco oscilatória, com baixo *overshoot*, baixo tempo de estabilização e etc. Em outras palavras, o que se procura é um comportamento de resposta do sistema adequado, que não é bem expresso pelo erro total. O uso desses índices que expressam o comportamento da resposta, também apresentam problema de generalidade, já que esses conceitos só são bem definidos para degraus.

Desse modo, a escolha da função custo é, sem dúvida, uma das dificuldades na otimização automática de controladores e um dos fatores mais importantes na obtenção da resposta desejada. Uma escolha inadequada da função custo pode levar o algoritmo de otimização a convergir para um resultado que supostamente seria o ótimo, por possuir a melhor função custo, que no entanto não constitui aquele realmente desejado. Diante disso, existe o desafio de se encontrar uma estratégia que possibilite uma forma mais genérica, adequada e próxima à humana de avaliar o desempenho de controladores, visando sua otimização, independentemente da forma do sinal de referência a ser utilizado. A solução que será aqui abordada é a utilização da transformada *wavelet*, que consiste em uma poderosa ferramenta matemática.

Kronland-Martinet *et al.* (1987) utilizaram transformadas *wavelet* para análise de padrões sonoros. Mallat (1989) mostrou os conceitos da transformada *wavelet*, apresentando como um sinal é decomposto em funções *wavelets* em diferentes resoluções de frequência, representando, então um sinal no domínio *wavelet*. O mesmo foi feito por Magalhães (2007). Vilani e Sanches (2013), utilizaram este recurso juntamente com a transformada de Fourrier para a análise aplicada à temperatura do ar. A análise *wavelet* permite a apreensão de informações do comportamento e forma do sinal. A capacidade de decompor o sinal na frequência e no tempo de forma simultânea, significa que se pode apreender a frequência de um sinal ao longo do tempo, característica que pode ser desejável, na avaliação e projeto de controladores, sendo assim, possível avaliar separadamente o desempenho do transitório e do regime permanente, visto que estes apresentam diferentes frequências. Assim, pode-se avaliar a suavidade e a acurácia da resposta.

Diante dessa capacidade da *wavelet* de decompor os sinais, se visa isolar descritores que expressam diferentes aspectos da resposta do sistema com o controlador, como seu comportamento no regime permanente, seu comportamento no transitório, se o ruído está sendo amplificado, entre outros, de modo que cada um desses aspectos seja um objetivo, constituindo-se então, um problema com vários objetivos. Acontece que, como na maioria dos problemas de otimização com vários objetivos, esses objetivos costumam ser conflitantes entre si. Desse modo, torna-se necessário utilizar técnicas de otimização multiobjetivo, capazes de lidar com esse problema. Essas técnicas, possuem um decisor que deve optar por uma solução que pondere os objetivos da forma mais adequada para o pretendido globalmente.

Nos problemas de otimização multiobjetivo, ao contrário dos de otimização mono-objetivo, não existem, usualmente, soluções que minimizam ou maximizam simultaneamente todos os objetivos. Quando todos os objetivos são de igual importância na otimização

multiobjetivo existe um conjunto grande de soluções aceitáveis que são superiores às demais denominadas soluções Pareto-ótimas ou eficientes. A escolha de uma dessas soluções depende das características do problema e será feita por um decisor.

De acordo com Arroyo *et al.* (2002), as técnicas de otimização utilizadas para solução de problemas de programação linear e não linear foram em sua maioria propostas até a década de 80. As técnicas utilizadas para programação linear multiobjetivo costumam usar métodos exatos de otimização mono-objetivo, nesse caso as soluções ótimas são encontradas resolvendo problemas derivados do original cujos ótimos correspondem a soluções ótimas globais. Essas técnicas não são adequadas para solucionar problemas mais complexos como os de otimização combinatória multiobjetivo, sendo usualmente problemas NP-completos, não podendo ser resolvidos através de algoritmos de tempo polinomial como aqueles destinados a resolver problemas de programação linear (ARROYO *et al.*, 2002).

No caso dos problemas NP-completos é improvável que se consiga desenvolver um algoritmo capaz de resolver o problema de forma exata em tempo razoável, assim, se procura obter uma solução de boa qualidade, uma solução inexata, porém, subótima em tempo compatível. As técnicas de otimização são capazes de transportar o mecanismo de adaptação para construir um procedimento computacional para tratar de problemas de otimização de alta complexidade (CUNHA R. TAKAHASHI, 2012), sendo classificadas em determinísticas e heurísticas. Os métodos determinísticos são baseados em valores numéricos exatos, como derivadas ou gradientes, ou aproximações destas; os heurísticos são estocásticos, no sentido que em diferentes execuções um método retornará diferentes soluções, porém tendem a uma solução subótima.

Os métodos heurísticos, especialmente as meta-heurísticas, foram o foco de inúmeras pesquisas na década de 90 visando resolver problemas de otimização multiobjetivo difíceis computacionalmente (EHRGOTT; GANDIBLEUX, 2000). Os métodos heurísticos podem ser divididos em três classes diferindo quanto à forma de explorar o espaço de soluções dos problemas: heurísticas construtivas, heurísticas de busca local e meta-heurísticas (ARROYO *et al.*, 2002).

As heurísticas construtivas produzem uma solução adicionando componentes por meio de regras específicas de acordo com a estrutura do problema. As heurísticas de busca local, também denominadas de heurísticas de busca em vizinhança, iniciam com uma solução e exploram a vizinhança desta solução através de regras de movimento que modificam a solução inicial até encontrar um ótimo local. As meta-heurísticas são métodos flexíveis que possuem a capacidade de incorporar novas situações, explorando de forma mais abrangente o espaço de soluções, evitando a estagnação em mínimos locais, embora não garanta a otimalidade global.

As meta-heurísticas se caracterizam pela aproximação com modelos naturais, sendo bastante utilizadas por basear-se em conceitos bastante simples e de fácil implementação, não necessitarem de informações de gradientes, podendo ignorar ótimos locais, e por serem aplicáveis em uma ampla gama de problemas. Existem diversos métodos de otimização propostos na literatura. De acordo com Alorf (2023), atualmente, o estado da arte em meta-heurísticas inclui algoritmos como o Genético, do Recozimento Simulado, da Evolução Diferencial, do Enxame de Partículas, da Colônia de Abelhas, da Colônia de Formigas,

da Busca Cuco, do morcego e do vaga-lume. O trabalho de Alorf (2023) apresentou uma revisão histórica sobre esses métodos.

As meta-heurísticas dividem-se em quatro tipos: as baseadas em processos evolucionários (*Evolution Based*), baseadas em processos físicos (*Physics Based*), baseadas em enxames (*Swarm Based*), e aquelas baseadas no comportamento humano (*Human Based*). As mais conhecidas e utilizadas, pela maior facilidade de implementação, são as *Evolution Based* e as *Swarm Based*. No que diz respeito a diminuição do custo computacional, as *Swarm Based* são mais eficientes, por utilizarem apenas operações matemáticas simples, quando comparados com etapas das *Evolution Based*, como seleção, *crossover*, mutação, o que as tornam mais simples de serem implementadas (MIRJALILI; LEWIS, 2016).

Como é comum que a otimização multiobjetivo resulte em problemas NP-completos, para gerar o conjunto das soluções Pareto-ótimas, os métodos heurísticos costumam ser os mais indicados para solucionar esses problemas, existindo variadas propostas de extensões de meta-heurísticas para esse fim (EHRGOTT; GANDIBLEUX, 2000; COELLO, 2000; VELDHUIZEN; LAMONT, 2000a; JONES *et al.*, 2002), sendo utilizadas para resolver tanto problemas multiobjetivos de otimização quanto problemas de otimização não linear.

Nesse contexto, entre as meta-heurísticas multiobjetivo, surgem os Algoritmos Evolucionários Multiobjetivo (MOEAs) nos quais se incluem os Algoritmos Genéticos Multiobjetivos (MOGAs). Os Algoritmos Genéticos (AGs) são técnicas de otimização inspiradas na evolução biológica, que, de forma simplificada, se baseiam em selecionar, recombinar e mutar uma população de soluções candidatas, representadas como cromossomos, buscando a "evolução" da solução mais adequada com fins a um determinado objetivo. Inicialmente prestando-se a resolver problemas com uma única função de custo ou aptidão, essas técnicas foram utilizadas como base para algoritmos destinados à solução de problemas com mais de um objetivo, os quais são denominados de MOGAs.

Os MOGAs são extensamente abordados na literatura, sendo aplicados em uma ampla diversidade de problemas, sendo frequentemente usados para otimizar sistemas complexos. Em sistemas de energia renovável, como a otimização de eficiência energética de turbinas eólicas (SHEN *et al.*, 2015).Em projeto de engenharia aeroespacial, na otimização do design de asas de aeronaves(TIAN; LI, 2020).No processos de manufatura, na otimização do processo de soldagem a laser (YANG *et al.*, 2018). Em redes de transporte, otimizando o sistema de transporte público (WITHERIDGE *et al.*, 2014).

Um dos desafios dos MOGAs é utilizar uma medida de desempenho para os múltiplos objetivos, algumas soluções encontradas são o indicador de hipervolume, que mede o volume do espaço objetivo dominado pelas soluções não dominadas e a distância geracional invertida, que avalia a distância entre as soluções não dominadas e a frente de Pareto estimada.

Uma variedade de MOGAs foram propostos, alguns dos mais populares são o Strength Pareto Evolutionary Algorithm (SPEA), desenvolvido por Zitzler e Thiele (1999), que utiliza a estimativa de densidade para garantir um conjunto de soluções não dominadas e o Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D), proposto por Zhang e Li (2007), que decompõe o problema em subproblemas de objetivo único e os otimiza simultaneamente.

Um dos primeiros e, possivelmente, mais populares famílias de MOGAs são os Non-

Dominated Sorting Genetic Algorithm (NSGA) desenvolvido por Deb *et al.* (2002), contando também com suas variações: o NSGA-II e NSGA-III. O NSGA opera mantendo um conjunto de soluções não dominadas, que são aquelas que não são dominadas por nenhuma outra solução na população. O NSGA se baseia em aplicar uma classificação a partir da dominância de Pareto para identificar as soluções não dominadas e realiza operações de seleção, cruzamento e mutação para evoluir a população.

Neste trabalho, será apresentada a utilização da análise *wavelet* para obtenção de objetivos a serem otimizados usando os algoritmos NSGA-II e NSGA-III. Para avaliar a estratégia proposta, será realizada a sintonia e otimização de controladores *fuzzy* para um sistema de tanques acoplados, para um sistema de pendulo invertido e para um motor DC, para controle de posição e de velocidade. No Capítulo 2, será apresentada a introdução. No Capítulo 3, será apresentada a metodologia proposta. No Capítulo 4, serão apresentados estudos de casos com seus respectivos resultados obtidos e o Capítulo 5 apresentará as conclusões.

Capítulo 2

Fundamentação Teórica

Neste capítulo serão apresentados conceitos essenciais para a compreensão do trabalho. Serão apresentados os tipos de controladores que serão otimizados. Serão introduzidas as meta-heurísticas mono-objetivo e as heurísticas de otimização multiobjetivo clássicas, técnicas que dão fundamento às técnicas de otimização multiobjetivo que serão utilizadas, que são as meta-heurística multiobjetivo, as quais serão apresentadas na sequência. Além disso, alguns índices de avaliação de controladores serão abordados e, por fim, será apresentada a transformada *wavelet*, que no próximo capitulo será utilizada para se propor uma metodologia de avaliação e otimização para controladores.

Assim, nesse capitulo, serão abordados controladores *fuzzy* na seção 2.1, Meta-heurísticas na seção 2.2, otimização multiobjetivo na seção 2.3, meta-heurísticas multiobjetivo na seção 2.4, incluindo NSGA-II e NSGA-III, índices de avaliação de controladores na seção 2.5 e transformada *wavelet* na seção 2.6.

2.1 Controladores *Fuzzy*

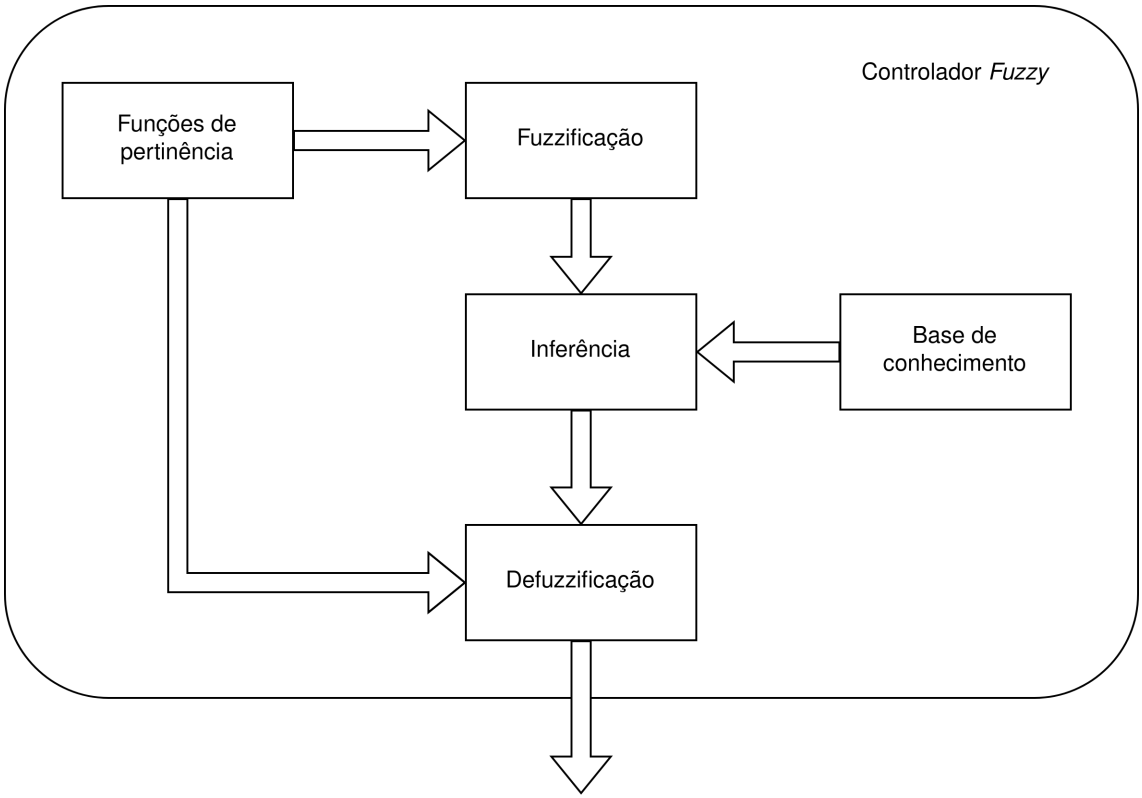
O controle *fuzzy*, é inserido no contexto da inteligência artificial, permitindo codificar o conhecimento e a experiência de operadores humanos e lidar com incertezas imitando o modo de tomar decisões do ser humano. A lógica *fuzzy*, utilizada nesse tipo de controlador é baseada na teoria dos conjuntos nebulosos de Zadeh (1965).

A lógica clássica se caracteriza por ser binária, um elemento sempre possui pertinência total ou não pertinência total a determinado conjunto. A esse tipo de conjunto, regido pela lógica clássica dá-se o nome conjunto *crisp*, caracterizado por só permitir dois valores, o de está contido ou o de não está contido. A teoria *fuzzy* surgiu devido à necessidade de tratar informações imprecisas presentes na linguagem humana, expressões do tipo “quase”, “um pouco”, “muito” não podiam ser representadas pela lógica clássica. Para isso a lógica *fuzzy* possibilita que um elemento tenha pertinência ou não pertinência parcial a um dado conjunto, este fato da uma maior plasticidade a conjuntos de regras e sistemas de inferências.

A flexibilidade possibilitada pela lógica *fuzzy* permite aos controladores *fuzzy* apresentar vantagem em lidar com incertezas e imprecisões, simulando a forma de raciocínio humana, tornando-se capaz de lidar com problemas complexos de controle linear e não linear. Essa flexibilidade por outro lado acaba demandando uma necessidade de mais co-

nhecimento agregado, devido a sua estrutura e lógica mais complexas e grande quantidade de parâmetros que podem ser ajustados. O projeto de um controlador fuzzy usualmente envolve a codificação da experiência adquirida por operadores humanos, seguido por um ajuste fino de um número usualmente grande de parâmetros, tornando a sintonia de controladores deste tipo difícil. Os blocos funcionais nos quais pode ser dividido um controlador *fuzzy* são ilustrados na Figura 2.1.

Figura 2.1: Os blocos funcionais em que se pode dividir um controlador *fuzzy*



Fonte: Autoria própria

O funcionamento do controlador *fuzzy* pode ser explicado de forma simplificada em três etapas: fuzzificação, inferência e defuzzificação. Em termos gerais a fuzzificação do sinal de entrada consiste na tradução dos valores de entrada nos conjuntos *fuzzy*. A inferência consiste numa lista de regras ativadas por uma dada combinação de entradas possíveis e fornecendo cada qual uma saída. O processo de converter essa saída do domínio *fuzzy* em um sinal de controle é denominada defuzzyficação.

A fuzzificação tem como função mapear números reais para o domínio *fuzzy* transformando os valores mapeados em variáveis linguísticas definidas pelas funções denominadas funções de pertinência. Essas são funções matemáticas que descrevem um universo de estudo, podem possuir diversos formatos, sendo comum a utilização de funções do tipo trapezoidal e triangular. A forma dessas funções resultam em parâmetros para a sintonia representando, por exemplo, as aberturas à direita, a abertura a esquerda e o centro no caso da função triangular. A forma das funções é geralmente dada pela experiência de um

especialista humano codificando o conhecimento adquirido por esse. A atribuição de um valor segundo as funções de pertinência estabelecem o grau de pertinência ao grupo que aquela função representa. Desta forma representa-se a entrada com sua pertinência a cada variável linguística no domínio *fuzzy*.

Esses dados resultantes da fuzzificação são submetidos à base de conhecimento, que agrega informações de dados e regras que são representadas na forma linguística de afirmações SE-ENTÃO, sendo uma lista das combinações de entrada possíveis, fornecendo regras para o processo de controle sem contradições entre si. Essa inferência da base de regras simula as decisões que seriam tomadas por humanos no caso um especialista. Quando uma regra é ativada pelas entradas fuzzificadas as correspondentes saídas *fuzzy* são ativadas em um determinado grau de pertinência.

Neste contexto a defuzzificação utiliza o resultado deste processo de inferência para convertê-lo em um sinal de controle capaz de atuar na planta. Ou seja a defuzzificação é responsável por converter valores do domínio *fuzzy* para valores reais a serem aplicados na entrada do sistema a ser controlado.

Dois modelos *fuzzy* se destacam: o Mamdani e o Takagi Sugeno. O modelo proposto por Mamdani e Assilian (1975) é utilizado quando possui-se conhecimento especialista sobre a planta que se deseja controlar, codificando as saídas de forma linguística, permitindo descrever o conhecimento humano de forma intuitiva. O modelo proposto por Takagi e Sugeno (1985) possui como saída uma linguagem mais voltada para representação matemática. Os dois modelos diferem na forma em que se calcula a saída do sistema.

A seguir serão apresentadas algumas arquiteturas comumente usadas para o controle utilizando *fuzzy*.

2.1.1 Controladores *fuzzy*-PID

Um controlador *fuzzy* pode ser projetado de modo análogo a controladores do tipo PID. Esta abordagem permite que controladores *fuzzy* comportem-se semelhante a controladores PID, que tem uma lógica mais familiar aos projetistas de controladores, porém com a capacidade de lidar com não linearidades e de simularem o conhecimento humano, consoante a Simões e Shaw (2007). Serão apresentados quatro tipos, *fuzzy*-P, *fuzzy*-PD, *fuzzy*-PI e *fuzzy*-PID sendo análogos respectivamente aos controles P, PD, PI e PID.

2.1.1.1 Controlador *fuzzy*-P

O *fuzzy*-P é definido apenas por uma entrada, o erro entre o sinal de referência e o sinal da variável observada, $e(t)$ e por uma saída $u(t)$, que representa o sinal de controle que será aplicado à planta. Assim, tem-se a analogia ao controlador P que pode ser calculado da seguinte forma:

$$u(t) = K_p \cdot e(t) \tag{2.1}$$

2.1.1.2 Controlador *fuzzy*-PD

O *fuzzy*-PD tem duas entradas, o erro entre o sinal de referência e o sinal da variável observada representado por $e(t)$ e $\frac{de(t)}{dt}$, que representa a variação do sinal do erro. Possui ainda uma saída $u(t)$, que é o sinal de controle que será aplicado à planta. Desse modo, obtém-se a analogia ao controlador PD que pode ser calculado da seguinte forma:

$$u(t) = K_p \cdot e(t) + K_d \cdot \frac{de(t)}{dt} \tag{2.2}$$

2.1.1.3 Controlador *fuzzy*-PI

O *fuzzy*-PI tem duas entradas, o erro entre referência e variável observada $e(t)$ e sua integral, $\int e(t)dt$, e ter-se-ia como saída o sinal de controle a ser aplicado $u(t)$. Apartir disso, pode-se obter a analogia ao controlador PI que pode ser calculado da seguinte forma:

$$u(t) = K_p \cdot e(t) + K_i \cdot \int e(t)dt \tag{2.3}$$

Pode-se, ainda, expressar o controlador *fuzzy*-PI, empregando como entradas o erro entre o sinal de referência e o sinal da variável observada representado por $e(t)$ e $\frac{de(t)}{dt}$ que representa a variação do sinal do erro. Possui ainda uma saída, $\frac{du(t)}{dt}$, que é a variação do sinal de controle que será aplicado à planta:

$$\frac{du(t)}{dt} = K_i \cdot e(t) + K_p \cdot \frac{de(t)}{dt} \tag{2.4}$$

Observa-se que a Equação 2.4 corresponde a derivada da Equação 2.3, o que é um artifício matemático para evitar trabalhar numericamente com a integral do erro, devido aos problemas inerentes a este tipo de sinal. Assim, na saída de um controlador *fuzzy*-PI deve ser associado um integrador, para que se possa obter o valor do sinal de controle.

2.1.1.4 Controlador *fuzzy*-PID

O *fuzzy*-PID tem três entradas, $e(t)$ é o erro entre o sinal referência e o sinal da variável observada, $\frac{de(t)}{dt}$, a variação do sinal do erro, $\int e(t)dt$, a integral do erro. Possui ainda uma saída , $\frac{du(t)}{dt}$ que é a variação do sinal de controle que será aplicado à planta. Assim, obtém-se a analogia ao controlador PID dada pela seguinte forma:

$$u(t) = K_p \cdot e(t) + K_i \cdot \int e(t)dt + K_d \cdot \frac{de(t)}{dt} \tag{2.5}$$

Pode-se empregar a estratégia de utilizar como entradas o erro entre referência e variável observada, $e(t)$, sua derivada $\frac{de(t)}{dt}$ e sua segunda derivada $\frac{d^2e(t)}{dt^2}$. Além disso, ter-se-ia como saída$\frac{du(t)}{dt}$ que é a variação do sinal de controle a ser aplicado à planta:

$$\frac{du(t)}{dt} = K_i \cdot e(t) + K_p \cdot \frac{de(t)}{dt} + K_d \cdot \frac{d^2e(t)}{dt^2} \tag{2.6}$$

Aqui se emprega mesma estratégia utilizada para obter o controlador *fuzzy*-PI, análoga ao controlador PI clássico expresso na Equação 2.4, a partir da Equação 2.3, também é empregada para obter a Equação 2.6 partindo da Equação 2.6. Também neste caso, deve ser associado um integrador à saída do controlador *fuzzy*, para que se possa obter o valor do sinal de controle.

2.2 Meta-heurísticas

Métodos de otimização são procedimentos matemáticos utilizados em diversas aplicações que maximizam uma função objetivo ou minimizam uma função custo, ou seja, um algoritmo que busca melhores resultados com base em uma função de avaliação escolhida de acordo com o problema.

Existem diversos métodos de otimização, divididos em métodos determinísticos, quando é possível prever onde todas as etapas terão seu ponto de partida, ou métodos estocásticos, quando são baseados em um processo aleatório, rodando iterativamente para encontrar a melhor solução. Entre os métodos estocásticos existem técnicas que são definidas por regras derivadas empiricamente a partir das quais melhores soluções são definidas. Destacam-se, nesse contexto, as heurísticas, que são algoritmos exploratórios projetados para resolver um determinado problema sem utilizar recursos extensos, não explorando todo o espaço de busca, não garantindo obter a melhor solução, ou seja, não garantindo otimalidade, mas buscando encontrar uma solução de boa qualidade. Normalmente, essas técnicas envolvem implementações computacionais que não lidam com conhecimento especializado do problema a ser otimizado, caso em que são chamadas de "buscas cegas", partindo de soluções factíveis e buscando aproximar o ponto ótimo por meio de iterações sucessivas.

Dentre as heurísticas, existe um grupo chamado de meta-heurísticas, que são estratégias inteligentes para projetar ou melhorar programas heurísticos. São aplicadas para encontrar respostas a problemas que carecem de informações, não se sabendo a forma da função a ser otimizada e havendo pouca informação heurística disponível, de sorte que métodos baseados em força bruta não são viáveis devido ao espaço de solução ser muito grande. Nesse contexto, segundo a definição original, meta-heurísticas são métodos de solução que coordenam procedimentos de buscas locais com estratégias de mais alto nível, de modo a criar um processo capaz de escapar de mínimos locais e realizar uma busca robusta no espaço de soluções de um problema (GLOVER; GREENBERG, 1989).

As meta-heurísticas normalmente se caracterizam pela aproximação com modelos naturais, sendo bastante utilizada por basear-se em conceitos bastante simples e de fácil implementação, não necessitando de informações de gradientes, podendo ignorar ótimos locais e tendo capacidade de serem utilizadas em uma ampla gama de problemas. As meta-heurísticas dividem-se em quatro tipos: as baseadas em processos evolucionários (*Evolution Based*), baseadas em processos físicos (*Physics Based*), baseadas em enxames (*Swarm Based*), e aquelas baseadas no comportamento humano (*Human Based*). As mais conhecidas e utilizadas, pela maior facilidade de implementação são as *Evolution Based* e as *Swarm Based*. No que diz respeito a diminuição do custo computacional, as *Swarm Based* são mais eficientes, por utilizarem apenas operações matemáticas simples, enquato

as *Evolution Based*, com suas etapas de seleção, *crossover*, mutação, são mais simples de serem implementadas (MIRJALILI; LEWIS, 2016). A tabela 2.1 apresentará algumas meta-heurísticas frequentemente encontradas na literatura.

Tabela 2.1: Quadro de meta-heurísticas

Meta-heusrística	Descrição	Referências
Particle Swarm Optimization (PSO)	PSO é uma meta-heurística baseada em população que simula o comportamento de um enxame de partículas em um espaço de busca. Cada partícula representa uma solução candidata e seu movimento é guiado por sua própria melhor posição anterior e pela melhor posição encontrada pelo enxame.	KENNEDY, James; EBERHART, Russell. Particle swarm optimization. In: **Proceedings of ICNN'95-international conference on neural networks**. IEEE, 1995. p. 1942-1948.
Simulated Annealing (SA)	SA é um algoritmo de otimização estocástica inspirado no processo físico de recozimento em metais. Ele começa com uma solução inicial e a modifica iterativamente, aceitando soluções piores com uma certa probabilidade com base em um parâmetro de temperatura que diminui com o tempo.	KIRKPATRICK, Scott; GELATT JR, C. Daniel; VECCHI, Mario P. Optimization by simulated annealing. **science**, v. 220, n. 4598, p. 671-680, 1983.
Tabu Search (TS)	TS é uma meta-heurística que mantém uma lista tabu de soluções visitadas recentemente para evitar revisitá-las no futuro. Ele modifica iterativamente a solução atual realizando movimentos que não são proibidos pela lista tabu, podendo aceitar soluções piores para escapar de ótimos locais.	GLOVER, Fred. Tabu search—part I. **ORSA Journal on computing**, v. 1, n. 3, p. 190-206, 1989. GLOVER, F.; LAGUNA, M. Tabu search. **Kluwer Academic Publishers**. 1997.

Ant Colony Optimization (ACO)	ACO é uma meta-heurística baseada em população, inspirada no comportamento de forrageamento de formigas. Ele usa uma colônia de formigas artificiais para pesquisar o espaço de solução, para o qual cada formiga representa uma solução candidata e trilhas de feromônio guiam a busca.	DORIGO, Marco; STÜTZLE, Thomas. **Ant colony optimization algorithms for the traveling salesman problem**. 2004. SOCHA, Krzysztof; DORIGO, Marco. Ant colony optimization for continuous domains. **European journal of operational research**, v. 185, n. 3, p. 1155-1173, 2008.
Algoritmo de Colônia de Abelhas Artificiais (ABC)	ABC é uma meta-heurística baseada em população que é inspirado no comportamento de forrageamento das abelhas. Ele usa uma população de abelhas artificiais que exploram o espaço de busca, modificando iterativamente as soluções candidatas e se comunicando entre si por meio de um processo de dança.	KARABOGA, Dervis; BASTURK, Bahriye. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. **Journal of global optimization**, v. 39, p. 459-471, 2007. KARABOGA, Dervis. Artificial bee colony algorithm. **scholarpedia**, v. 5, n. 3, p. 6915, 2010.
Gray Wolf Optimizer (GWO)	GWO é um algoritmo meta-heurístico baseado em população inspirado no comportamento de caça de lobos cinzentos. Ele usa uma população de lobos artificiais que são organizados em matilhas e modificam iterativamente as soluções candidatas com base em três tipos diferentes de comportamento de caça: procurar, cercar e atacar a presa.	MIRJALILI, Seyedali; MIRJALILI, Seyed Mohammad; LEWIS, Andrew. Grey wolf optimizer. **Advances in engineering software**, v. 69, p. 46-61, 2014.

Evolução Diferencial (DE)	DE é um algoritmo de otimização baseado em população, envolvendo a criação de uma população de soluções candidatas e, em seguida, a melhoria iterativa da população usando uma combinação de operações de mutação e cruzamento.	STORN, Rainer; PRICE, Kenneth. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. **Journal of global optimization**, v. 11, n. 4, p. 341, 1997.
Cuckoo Search (CS)	CS é um algoritmo meta-heurístico baseado em população que é inspirado no comportamento dos cucos. O algoritmo envolve a criação de uma população de soluções candidatas e, em seguida, melhorando iterativamente a população usando uma combinação de estratégias de busca local e global.	YANG, Xin-She; DEB, Suash. Cuckoo search via Lévy flights. In: **2009 World congress on nature & biologically inspired computing (NaBIC)**. Ieee, 2009. p. 210-214.
Gravitational Search Algorithm (GSA)	GSA é um algoritmo de otimização baseado em população inspirado na lei da gravidade e no comportamento de objetos celestes. O algoritmo envolve a criação de uma população de soluções candidatas e, em seguida, melhorando iterativamente a população usando uma combinação de forças de atração e repulsão entre as soluções.	RASHEDI, Esmat; NEZAMABADI-POUR, Hossein; SARYAZDI, Saeid. GSA: a gravitational search algorithm. **Information sciences**, v. 179, n. 13, p. 2232-2248, 2009.

Fonte: Autoria própria

O Algoritmo Genético e o Enxame de Partículas são as meta-heurísticas mais frequentemente abordadas na literatura quando se trata de algoritmos evolutivos e de enxames, respectivamente(MIRJALILI *et al.*, 2014b). Nesses métodos, usam-se técnicas de heurísticas e procedimentos probabilísticos combinados como guia na busca em todo espaço de pesquisa e sua vizinhança, evitando paradas prematuras em mínimos locais.

2.2.1 Algoritmos Evolutivos (EAs)

Os algoritmos evolutivos (EA) fazem parte da Computação Evolutiva se constituindo em técnicas promissoras de meta-heurísticas que se tornaram um expoente em pesquisa na área de otimização nas últimas décadas (VIKHAR, 2016). Graças a sua natureza flexível e de comportamento robusto herdado da computação evolutiva, torna-se uma maneira eficiente de resolver problemas de otimização globais generalizados, sendo utilizado com sucesso em muitas e complexas aplicações (VIKHAR, 2016)(YAO, 1997).

A computação evolutiva inclui muitos algoritmos diferentes, comumente conhecidos como algoritmo evolutivo (EA) (EIBEN; SMITH, 2015)(AL-SALAMI, 2009). A EA usa simulação de evolução biológica para descobrir soluções para problemas complexos do mundo real. A evolução é mais adequada para aplicações em que as heurísticas clássicas, não podem ser utilizadas ou apresentam resultados insatisfatórios. Um EA inspirado nos conceitos de evolução biológica, como reprodução, mutação, recombinação e seleção. Nesses algoritmos se utilizam estratégias evolutivas sob um conjunto de soluções candidatas e produz-se um novo conjunto de soluções buscando que essas apresentem um melhor desempenho seguindo alguma função que meça a adaptabilidade do indivíduo. Este é um processo iterativo continuado até que um candidato de qualidade suficientemente boa seja encontrado (CAMPOS; TEIXEIRA, 2006)(EIBEN; SMITH, 2015).

Os EAs apresentam algumas vantagens frente a outras técnicas de otimização (VIKHAR, 2016)(AL-SALAMI, 2009)(WHITLEY, 2001). Pode-se citar ser baseado em conceitos simples e flexíveis, por ser inspirado na evolução natural, o que facilita a implementação e a utilização para uma ampla gama de problemas. Outra vantagem é poder utilizar informações prévias, sem, contudo, estarem limitados a valores contínuos. Além disso, é possível que cada avaliação seja executada em paralelo, sendo necessário apenas que sejam realizadas operações durante o processo de seleção que requerem processamento serial, o que torna esses algoritmos com grande potencial de paralelização (VIKHAR, 2016). Os EAs podem tanto utilizar conhecimentos a priori aumentando a sua eficiência, quanto demonstram capacidade de resolver problemas sem qualquer experiência humana prévia (VIKHAR, 2016). Contudo, Vikhar (2016) aponta como algumas das principais desvantagens desses algoritmos não garantir sempre a solução ótima para um determinado problema dentro de um período de tempo determinado, e a necessidade de ajuste dos parâmetros do algoritmo.

De (JONG *et al.*, 1997) dividiu algoritmo evolucionário como programação evolutiva, estratégias evolutivas, programação genética e algoritmo genético. O Algoritmo Genético (GA) (GOLDBERG, 1989)(HOLLAND, 1975) é o tipo mais comum de EA (VIKHAR, 2016), apresentando de forma mais clara a simulação computacional de processos evolutivos naturais. Baseia-se em utilizar cruzamento e mutação para encontrar uma solução para problemas, representando as soluções como cromossomos, em geral por uma cadeia de números binários, uma sequência de bits que representam os genes. O cruzamento no GA clássico se dá pela troca de bits, aplicado com determinada probabilidade e a mutação se dá pela inversão de bits, aplicada a cada bit com uma dada probabilidade, seleção na nova geração por roleta (EIBEN; SMITH, 2015). Existem outras versões desse mecanismo de evolução, ou seja, variação nos operadores evolutivos. Este algoritmo será mais detalhado na seção seguinte.

Por sua vez, a Estratégia Evolutiva (RECHENBERG, 1973), proposta pela primeira

vez em 1973 por Rechenberg visa a otimização de funções complexas, multimodais e não diferenciáveis (VIKHAR, 2016). Essa técnica representa as soluções por vetores de parâmetros com números reais, originalmente discretizados, usa mutação gaussiana, não utiliza seleção mas a estratégias de substituição $(\mu+,\lambda)$ o *crossover* originalmente não era utilizado e quando implementado é feito trocando componentes ou fazendo uma recombinação linear em alguns componentes (EIBEN; SMITH, 2015). Outra característica é o uso de taxas de mutação auto-adaptativas, visando adicionar variabilidade aos parâmetros dos genótipos fazendo com que os parâmetros individualmente evoluam (EIBEN; SMITH, 2015)(VIKHAR, 2016). Essa técnica vem sendo aplicada, por exemplo, em roteamento de redes, Bioquímica, Óptica (VIKHAR, 2016)(COELLO, 2005).

A Programação Evolucionária, por seu turno, foi proposta por Fogel (1966), inicialmente voltada ao ramo da inteligência artificial, lidando com autômatos de estado finito para aprendizado de máquina (EIBEN; SMITH, 2015)(VIKHAR, 2016). A representação e os operadores foram especializados para esta área de aplicação. Por esta técnica cada pai dá origem por mutação a um filho, e uma estratégia de substituição positiva foi usada (EIBEN; SMITH, 2015). Porém, atualmente existem estratégias de programação evolucionária, que passaram a incluir qualquer forma de representação e estratégias de evolução, se caracterizando por usar estratégia de substituição e por não usar cruzamento (EIBEN; SMITH, 2015). Normalmente se usa auto-adaptação de mutação gaussiana quando se utilizam vetores de valor real para representação de soluções. Outra característica é que esse algoritmo permite que os parâmetros numéricos cresçam (VIKHAR, 2016)(COELLO, 2005).

Já a Programação Genética, desenvolvida por Koza *et al.* (1992), normalmente utiliza a representação por árvores sintáticas de expressões lógicas formais e utiliza de cruzamento e mutação, dos AGs, adaptados para trabalhar em árvores com tamanhos variados. Sua função *fitness* indica a capacidade desse programa de resolver problemas computacionais (VIKHAR, 2016). Por isso, muitas vezes se considera que esta técnica trata-se de uma GA com representação em árvore. Porém, também se considera Programação genética quando se usa como representação expressões sintáticas, que podem ser vistas como programas, de tal modo que o que define essa técnica é estar destinada à evolução automática de programas.

Na prática, a maior diferença entre essas técnicas é a forma de representar o problema a ser otimizado. Assim, a melhor maneira de escolher uma estratégia específica para resolver um problema é escolher uma representação apropriada ao problema e posteriormente escolher operadores que evoluam representação correspondente ao problema(VIKHAR, 2016). Assim, pode-se definir qual técnica utilizar. Deve-se ainda ressaltar que essas técnicas contam com variações, apresentando diferentes operadores, que tornam mais eficientes para determinados problemas, o que também pode ser levado em consideração na decisão de que técnica será utilizada.

Também é importante citar duas extensões aos algoritmos evolutivos, buscando aumentar sua eficiência, que são os algoritmos meméticos e os AEs distribuídos. Os Algoritmos Meméticos (FOGEL, 1966) aplicam a busca local para indivíduos de modo a combinar EA com busca local. Os pesos entre a busca local e o método EA podem variar aleatoriamente (EIBEN; SMITH, 2015). Os AEs distribuídos, visam superar a necessidade de recursos computacionais, para isso, busca adaptações aos algoritmos que permitam a

distribuição das tarefas computacionais, podendo distribuir as operações evolutivas entre vários computadores ou processadores, de modo que possam fazer os cálculos em paralelo. Esses algoritmos ao se utilizarem do paralelismo reduzem o tempo de processamento e aumentam a qualidade da solução encontrada no espaço de busca (VIKHAR, 2016).

Outra tendência, que visa alcançar melhores resultados por algoritmo evolutivo, é a hibridização de EAs com outros algoritmos existentes (VIKHAR, 2016). Pode ser citado o Genetic Swarm Optimization (GSO) (CHAKRADEO *et al.*, 2014), que combina GA com o enxame de partículas e o Hybrid PSO (CHAKRADEO *et al.*, 2014), que aplica mutação à técnica enxame de partículas para que não haja a estagnação em mínimos locais. Ademais, tem-se o Immune self-adaptive differential evolution (ISDE) (SUN *et al.*, 2012), que utiliza mecanismo de processamento de informações de sistemas imunológicos biológicos para adaptação de fatores de escala e cruzamento de EA e o Dynamic multi-agent GA (LIU *et al.*, 2015), que une a técnica de dynamic multi-agent com o algoritmo genético.

2.2.1.1 Funcionamento de um EA

O projeto de um EA se inicia com a escolha da representação de soluções candidatas. As soluções candidatas são vistas como fenótipos que podem ter estruturas muito complexas, por isso, esses fenótipos são representados por genótipos correspondentes. Assim, o mecanismo de funcionamento padrão da EA consiste em aplicar alguns operadores que geram variações nesses genótipos que permitem a busca orientada no espaço de busca, que está representado em forma de genes. Essas operações podem variar dependendo da estrutura de representação escolhida. De modo que o que acaba caracterizando uma técnica de EA é a representação. Outro ponto essencial é definir a função de aptidão ou de avaliação. É a aptidão que orienta a busca, levando à melhoria da qualidade fenotípica de acordo com os parâmetros que ela define, por isso a extrema importância de seu projeto.

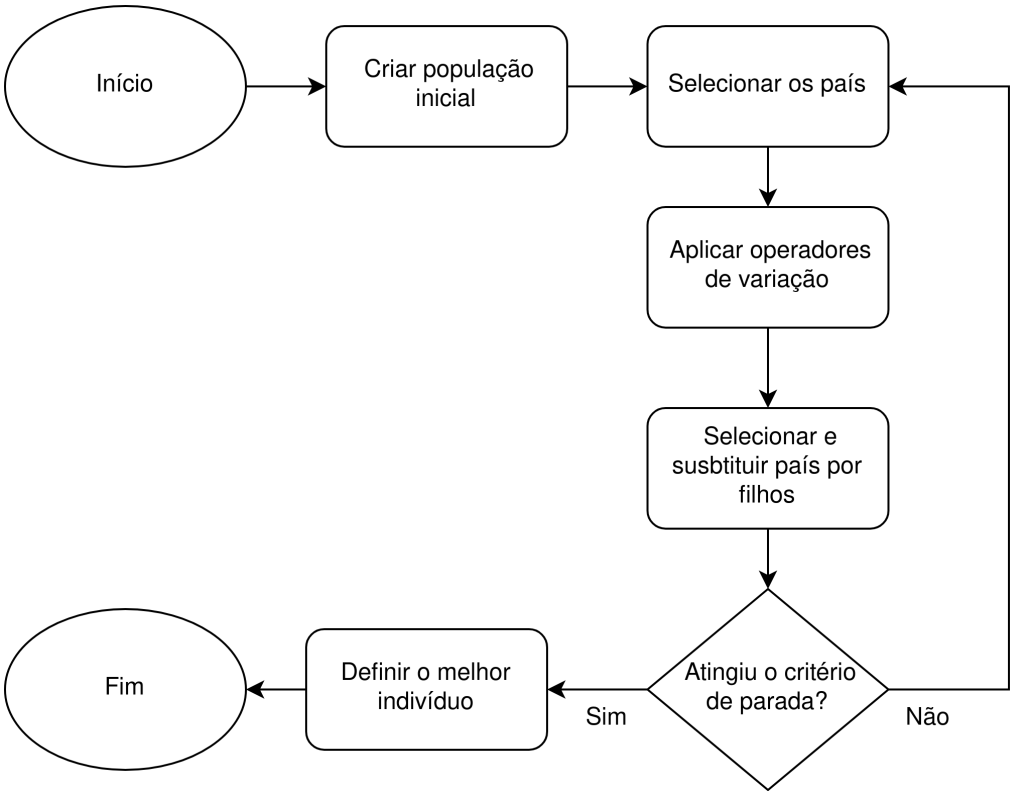
Após definidas a representação e a estrutura algorítmica de operações que levaram a evolução da população de soluções candidatas, deve-se definir a população inicial, que normalmente é criada através de uma amostra aleatória do espaço de busca, buscando, no geral, que ela seja produzida da maneira mais uniforme possível. Independente da representação, todo algoritmo evolucionário contará com operadores de seleção. Isso porque esses algoritmos se baseiam na teoria de Darwin, que afirma que os indivíduos com maior aptidão se reproduzem e sobrevivem. Por isso, o mecanismo básico de funcionamento de uma EA sempre consistirá em duas etapas de seleção, na primeira alguns pais são escolhidos para se tornarem genitores e em uma segunda etapa, após aplicar operadores de variação, há a substituição de alguns, ou todos, os pais por filhos, simulando artificialmente processo de seleção natural e evolução. Existem variadas formas de se realizar essas duas etapas dependendo do algoritmo evolucionário adotado.

Entre essas etapas de seleção, aplica-se sobre a população operadores de variação à depender da representação, que produzam soluções filhos a partir das soluções pais. No geral, esses operadores consistem em cruzamento e mutação. Enquanto os AGs utilizam o cruzamento essencialmente e a mutação é opcional e ocupa um espaço secundário, técnicas como a programação evolutiva não utilizaram cruzamento, mas apenas mutação.

Os operadores de cruzamento se baseiam em unir características de pais com uma boa aptidão esperando que a combinação de seus materiais genéticos gerem filhos com uma

aptidão ainda melhor. Existe uma série de estratégias para fazer essa recombinação de materiais genéticos a depender do algoritmo, como a combinação linear ou a troca de bits em determinadas posições. Os operadores de mutação, por sua vez, aplicam transformações estocásticas nos indivíduos, visando uma maior e melhor exploração no espaço de busca, mas sem causar grandes variações genotípicas, que resultariam em filhos muito distantes de seus pais, o que transformaria o algoritmo em uma busca burra. O processo de aplicação de operadores de seleção e de variação são realizados até que se atinja um critério de parada determinado. Seguindo esse algoritmo de funcionamento, pode-se apresentar o fluxograma de um EA conforme a Figura 2.3.

Figura 2.2: Fluxograma de funcionamento de um EA.



Fonte: Autoria própria

2.2.2 Algoritmo Genético (GA)

Holland (1975) propôs um novo conceito de algoritmo, o GA, classificado como um método evolutivo, uma vez que se baseia na teoria da evolução das espécies de Charles Darwin, capaz de explorar informações prévias para encontrar valores em seu espaço de busca, no qual se espera que os valores ótimos estejam presentes. O GA usa conceitos da genética e da evolução darwiniana como inspiração para realização de um algoritmo matemático, que funciona de forma análoga a esses processos biológicos. Ao se comparar o conjunto de possíveis respostas com a população de uma determinada espécie em evolução,

os indivíduos mais aptos tem maiores chances de propagarem seus genes ao longo das gerações e terem seus materiais genéticos conservados e recombinados, gerando indivíduos mais evoluídos, ou seja, mais aptos.

Segundo Bueno (2009), os algoritmos genéticos são baseados em um conjunto de soluções possíveis, não envolvem modelagem do problema, o algoritmo apresenta como resultado uma população de soluções e não apenas uma, trata-se de um método probabilístico, ou seja, uma mesma população inicial dificilmente apresentará os mesmos resultados para um mesmo problema.

Para a execução do algoritmo, cria-se uma população de indivíduos, em que cada indivíduo é uma solução a ser testada no problema. Essa população é avaliada de acordo com uma função objetivo e definida de acordo com o problema. Geralmente, cada indivíduo é codificado como um cromossomo, tendo suas informações a serem otimizadas dispostas como um vetor. Realiza-se a seleção, em que, por meio do índice de aptidão e baseado na avaliação, é encontrada a probabilidade de determinado indivíduo se reproduzir. O índice de aptidão indica qual dos candidatos tem maior chance de se reproduzir e passar seus "genes"para as próximas gerações, de modo a privilegiar os indivíduos mais aptos. A seleção pode ser usada tanto na escolha de quais indivíduos serão progenitores, quanto na escolha dos melhores adaptados para passar à próxima geração (SANTOS, 2002).

Após a seleção, serão escolhidos dois indivíduos para realizar o *crossover* ou cruzamento, que é a recombinação do material genético destes indivíduos, resultando em dois novos indivíduos, o que será repetido até se gerar uma nova população. Os indivíduos podem ser escolhidos para cruzamento segundo diversas estratégias. Pode-se utilizar o método da roleta, em que se irá sortear dois indivíduos para cruzamento, sendo a chance de ser escolhido proporcional a sua aptidão. Existem ainda outras estratégias na literatura, métodos como o rodeio, em que se formam grupos de indivíduos. Dentro deste grupo, o indivíduo será escolhido para realizar o cruzamento, tendo sempre mais chances de ser selecionado aquele com melhor desempenho. Na estratégia adotada por Bueno (2009), para cada cruzamento são escolhidos, aleatoriamente, dois indivíduos já previamente selecionados. Em seguida, são definidos, também aleatoriamente, dois locus para então as cargas genéticas serem trocadas gerando dois novos indivíduos.

Existem também diversos métodos de cruzamento ou *crossover*. Este pode ser implementado pela simples troca de parte de um cromossomo com o outro em um determinado ponto, que pode ser aleatória ou não. Existem métodos matemáticos que operam com base na média aritmética e diversos outros métodos, geralmente provenientes da combinação dessas estratégias, no que tange a técnicas de cruzamento. Alguns indivíduos podem perpetuar seu material genético sem a necessidade de serem escolhidos para cruzamento, essa técnica é denominada elitismo e se escolhem os melhores indivíduos da população para terem seus materiais genéticos conservados na próxima população.

Além do cruzamento pode ser aplicada a técnica de mutação, em que o indivíduo selecionado pode sofrê-la ou não. Para realização de mutação, inicialmente é definida a taxa, ou porcentagem de mutação na população, com base na qual é dada a probabilidade de um indivíduo sofrer ou não mutação. No geral, a mutação altera um indivíduo de forma aleatória, permitindo que o espaço de busca seja varrido de forma mais eficiente e completa. Segundo Bueno (2009), a parametrização da mutação e de sua taxa de ocorrência permite

um maior controle sobre a diversidade e evolução da população, o que pode ser desejável quando se almeja estabelecer parâmetros otimizados de busca.

Após a geração da nova população é feita novamente uma avaliação e pode-se aplicar o *crossover* repetindo o processo até que seja atingido um critério de parada pré-definido. O critério de parada não garante solução ótima, baseando-se em soluções quase ótimas. Apesar das diferentes metodologias adotadas pelos diversos autores para implementação de algoritmos genéticos, a estrutura base desses algoritmos permanece a mesma. Pode-se observar na Figura 2.3 o fluxograma de um AG.

Assim, o GA une a utilização de números gerados aleatoriamente e informações de gerações anteriores para avaliar e melhorar uma população de potenciais candidatos ao invés de um único ponto de cada vez (MARLER; ARORA, 2004), aproveitando-se do fato de existir uma população de valores a serem testados, e não apenas um único ponto, tornando mais fácil a busca por todo espaço de pesquisa, além de empregar operações como a mutação, que garante que o algoritmo não estacionará em pontos de mínimo ou máximo local.

2.3 Otimização Multiobjetivo

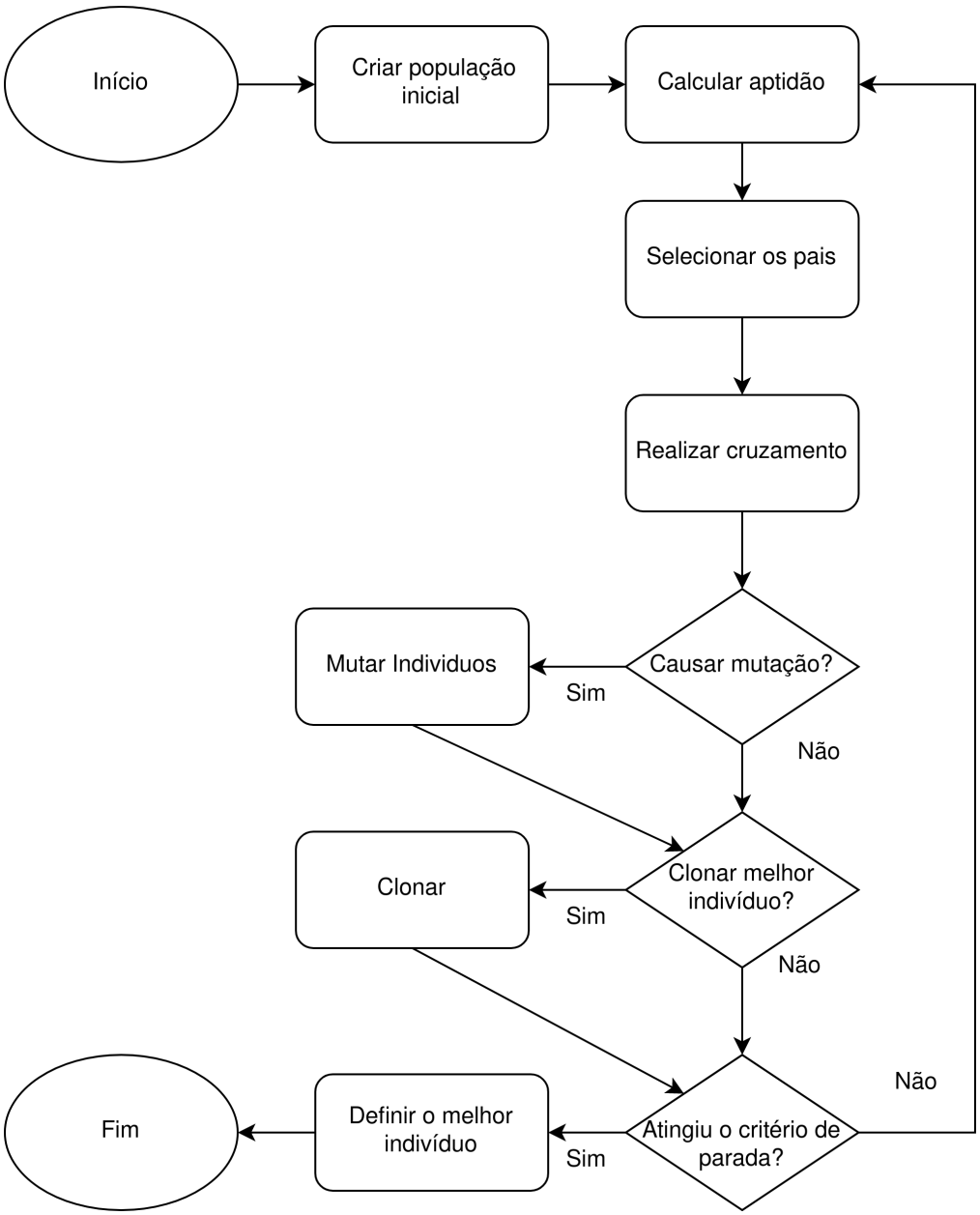
Um problema de otimização multiobjetivo é a busca de um conjunto de variáveis que resultem em uma solução capaz de satisfazer algumas restrições e otimizar uma função constituída por diversas funções objetivo ou restrições. Para tanto deve-se minimizar, ou maximizar dependendo do caso, um grupo de critérios e satisfazer um conjunto de restrições de forma simultânea. No caso multiobjetivo não se pode apontar uma solução única que otimize cada um dos critérios, mas um conjunto de soluções eficientes, denominadas Pareto-ótimas, no qual nenhuma solução é melhor que outra solução para todos os objetivos (ARROYO *et al.*, 2002). Nesse contexto é o decisor que escolherá uma das soluções Pareto-ótima que pondere os objetivos globais de um dado problema. Nesta sessão serão apresentados os conceitos básicos, a formulação matemática de problemas de otimização multiobjetivo e métodos clássicos de obtenção de soluções Pareto-ótimas e de avaliação de algoritmos heurísticos.

2.3.1 Conceitos Básicos

Na otimização multiobjetivo, busca-se encontrar uma solução composta por vetor de variáveis de decisão que satisfaça restrições e otimize uma função vetorial cujos elementos são as funções objetivo. Estas funções representam os critérios de otimalidade que, usualmente, são conflitantes, encontrando soluções que resultem em valores dos objetivos que não podem ser melhorados simultaneamente(ARROYO *et al.*, 2002).

Isto pode ser definido como:

Figura 2.3: Fluxograma de funcionamento de um AG.



Fonte: Autoria própria

$$
\begin{gathered}
\text{Minimizar (ou maximizar) } z = f(x) = (f_1(x), f_2(x), ..., f_r(x)) \\
\text{Sujeito a } x \in X^* \\
onde, \\
X = \text{Espaço de Decisões;} \\
x = (x_1, x_2, ..., x_n) \in X; \\
Z = \text{imagem de } X \text{ ou Espaço Objetivo;} \\
z = (z_1, z_2, ..., z_r) \in Z; \\
X^* = \{x \in X : g(x) \leq b\} = \text{Conjunto de Soluções Factíveis;} \\
Z^* = \{f(x) : x \in X^*\} = \text{Espaço Objetivo Factível;} \\
g(x) = \text{restrições;} \\
b \in R^p.
\end{gathered}
\tag{2.7}
$$

Sendo $f(*)$ uma função objetivo, o espaço objetivo factível é completamente ordenado para o caso mono-objetivo, ou seja, para qualquer dois elementos $x, y \in X^*$ sempre ou $fx) \leq f(y)$, ou $f(x) \leq f(y)$. Assim, a otimização buscará encontrar a solução (ou soluções) que forneça(m) o mínimo (ou máximo) valor de $f(*)$. No caso multiobjetivo, nos quais normalmente se considera vários objetivos conflitantes, não existe uma única solução que seja ótima com respeito a todos os objetivos, de modo que, minimizar (ou maximizar) um dos objetivos pode causar o incremento (ou decremento) dos demais. Assim sendo, o espaço objetivo, geralmente, não é completamente ordenado, mas é parcialmente ordenado como demonstrado por Pareto (1896). Um espaço de objetivos é parcialmente ordenado quando, para dois vetores de decisão quaisquer $x, y \in X^*$, há três possibilidades para os seus correspondentes vetores objetivos:

Ou $f_i(x) \leq f_i(y) : \forall i$, caso em que se define que o vetor objetivo $z_1 = f(x)$ domina $z_2 = f(y)$ e o vetor de decisão x domina y;

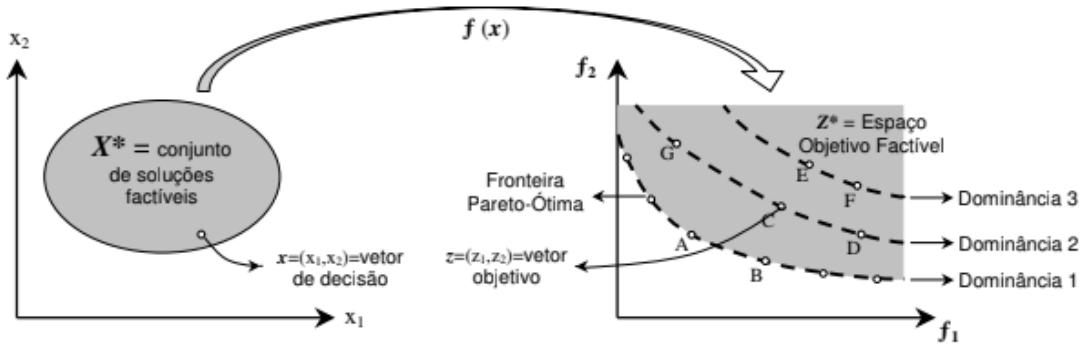
Ou $f_i(y) \leq f_i(x) : \forall i$, caso em que se define que o vetor objetivo $z_2 = f(y)$ domina $z_1 = f(x)$ e o vetor de decisão x é dominado por y;

Ou, ainda, $f_i(x) \nleq f_i(y)$ e $f_i(y) \nleq f_i(x) : \forall i$, caso em que se define que o vetor objetivo $z_1 = f(x)$ é indiferente com $z_2 = f(y)$ e vice-versa, e que o vetor de decisão x é indiferente com y e vice-versa.

É o fato dos vetores objetivos apenas parcialmente ordenados que torna os problemas de otimização multiobjetivos complexos.

Uma solução $x \in X^*$ é dominada por um subconjunto $A \subset X^*$ se existe uma solução $y \in A$ tal que y domina x.

Figura 2.4: Conjunto de soluções factíveis, espaço objetivo factível e grau de dominância em um problema de minimização com dois objetivos.



Fonte: (HASHIMOTO, 2004)

Na Figura 2.4, que representa a otimização de dois objetivos $f_1(x)$ e $f_2(x)$, o ponto A oferece um valor menor para f_1 em comparação com B, porém B apresenta um valor menor para f_2, se comparada com o ponto A, de modo que, a redução do valor de um objetivo implica no aumento do outro. No exemplo da Figura 2.4 os pontos A e B dominam G, C e D, os pontos E e F são dominados por G, C e D, os pontos D e G são indiferentes com C, assim como, A é indiferente a B e E é indiferente a F.

Assim, pode-se definir a otimalidade de Pareto. A solução $x^* \in X^*$ será uma solução eficiente ou solução Pareto-ótima se não existe qualquer outra solução $x \in X^*$ que domine x^*. Já o ponto $z^* = f(x^*)$ pertencente ao espaço objetivo factível Z^* é denominado de ponto eficiente ou ponto Pareto-ótimo. O conjunto de todas as soluções eficientes é denominado conjunto eficiente ou conjunto Pareto-ótimo, enquanto a imagem em Z^* do conjunto eficiente é denominada fronteira Pareto-ótima. Na Figura 2.4 os pontos A e B são são os pontos Pareto-ótimos pertencendo à fronteira de Pareto.

É importante ainda definir o conceito de ponto ideal ou ponto utópico que é um ponto $z^o \in Z^*$ que $z_j^o = min\{f_j(x) : x \in X^*\}, j = 1, ..., r$

2.3.2 Métodos de Otimização Multiobjetivo

A solução dos problemas de otimização multiobjetivo se dá pela determinação do Conjunto Eficiente, um Subconjunto desse ou ainda, conjuntos de soluções próximas da Fronteira Pareto-Ótima. Para tanto se utiliza um tomador de decisão que define os critérios de busca do algoritmo de otimização pode ser feito combinando os objetivos do problema em um único objetivo, segundo pesos que ponderam a preferência do Tomador de Decisão (KAGAN, 1993), nesse caso se constituindo na otimização de um único objetivo podendo-se utilizar estratégias mono objetivo. Outra opção é classificar os objetivos em ordem otimizando o primeiro objetivo, sem considerar as demais, e seguindo para a otimização dos objetivos seguintes sucessivamente, considerando o valor ótimo anterior, estratégia que não garante a obtenção da solução Pareto-ótima. O tomador de decisão pode promover uma intervenção durante o processo de busca para guiar a busca para as regiões que sejam mais convenientes segundo o seu critério, o que pode ser feito por meio de um sistema especialista.

2.3.3 Classificação de Métodos Multiobjetivos

A solução de problemas multiobjetivos é realizada por busca de soluções e tomada de decisões (ARROYO *et al.*, 2002). O processo de busca, devido a complexidade, usualmente não pode ser realizado por métodos exatos. O processo de tomada de decisões utiliza critérios para a escolher a solução do conjunto Pareto-ótimo ou a proximidade de um ponto a ele. Quanto ao decisor os métodos de otimização multiobjetivos podem ser classificados em métodos a *priori*, a *posteriori* e iterativos (ARROYO *et al.*, 2002).

Os métodos a *priori* se caraterizam pela atuação do decisor antes do processo de busca. Isso pode ser implementado combinando os objetivos do problema em um único objetivo, determinando explicitamente pesos que reflitam a preferência por cada objetivo. Utilizando métodos de estratégias de busca mono-objetivo. Outra possibilidade de métodos a priori é classificar os objetivos em ordem decrescente de prioridade. O problema é então resolvido para o primeiro objetivo sem considerar os demais, segue-se resolvendo para os objetivo seguintes, sucessivamente considerando apenas os valores anteriores.

Os métodos a posteriori realizam o processo de decisão logo após a busca de soluções. A busca é feita considerando todos os objetivos com igual prioridade, a partir deste conjunto, o decisor seleciona a solução mais adequada. Nos métodos iterativos, o decisor

interfere durante o processo de busca guiando a busca para regiões mais promissoras.

Existem vários métodos de busca para otimização, que se classificam em métodos determinísticos, quando se pode prever todos os seus passos de posse do seu ponto de partida, ou estocásticos, quando se baseiam em processos aleatórios, executando-se de forma repetida para achar a melhor solução. Existem ainda técnicas definidas por uma regra derivada da experiência, definindo uma melhor solução baseada em suas regras. Essa solução não possui garantia de otimalidade, mas pode ser considerada aceitável.

A busca de soluções eficientes não é uma tarefa simples, em especial quando envolve funções não-lineares e elevado o número de dimensões, por onde realizar a busca, o que usualmente acontece. Assim, a busca exaustiva realizada pelos métodos tradicionais de programação linear e não-linear baseadas em encontrar soluções analíticas exatas, ou percorrer todo o espaço de busca em sua integralidade podem encontrar com precisão o conjunto Pareto-ótimo, porém podem tornar-se inviáveis de serem realizadas em tempo hábil, dependendo da complexidade do problema.

Como problemas de otimização multiobjetivos são usualmente NP-complexo, outra possibilidade que se apresenta são as heurísticas. Métodos heurísticos são algoritmos exploratórios, que têm como objetivo resolver um dado problema, sem que para isso seja necessário utilizar uma quantidade muito grande de recursos, buscando encontrar uma solução de boa qualidade, não necessariamente a ótima. Geralmente, não envolvem a implementação computacional de um conhecimento especializado, sendo nesse caso, denominado de "busca cega" e partem de uma solução viável, e por meio de iterações sucessivas buscam se aproximar de um ponto ótimo. As heurísticas podem ser classificadas em construtivas, busca local e meta-heurísticas.

As heurísticas construtivas são métodos de busca que constroem uma solução a partir de regras baseadas nas informações do problema, são algoritmos específicos que codificam o conhecimento de um especialista de como otimizar a solução do problema. Os métodos de busca local ou busca em vizinhança visam melhorar uma solução viável, são mais genéricos e procuram mapear a vizinhança de uma solução através de um critério estocástico ou combinatório. Muitas vezes, são necessários métodos mais flexíveis, as meta-heurísticas apresentam um importante papel pela sua capacidade de resolver problemas complexos. Em geral, uma meta-heurística é implementada por uma estrutura genérica baseada em princípios ou conceitos simples normalmente baseadas na natureza, sobreposta a uma heurística específica de otimização dessa estrutura (HASHIMOTO, 2004).

A seguir se apresentará as técnicas tradicionais de otimização multiobjetivo. Em seguida, serão brevemente abordadas as meta-heurísticas pela sua importância na sintonia de controladores.

2.3.4 Métodos Tradicionais de Otimização Multiobjetivo

Os métodos clássicos de solução de problemas de otimização multiobjetivo escalarizam os objetivos formando um único objetivo (ARROYO *et al.*, 2002). Assim, é criado um problema substituto, que transforma a otimização vetorial em um problema de otimização escalar. Os métodos clássicos são o da Soma Ponderada (COHON, 2004) e o do ε-restrito (STEUER; STEUER, 1986).

2.3.4.1 Método da Soma Ponderada

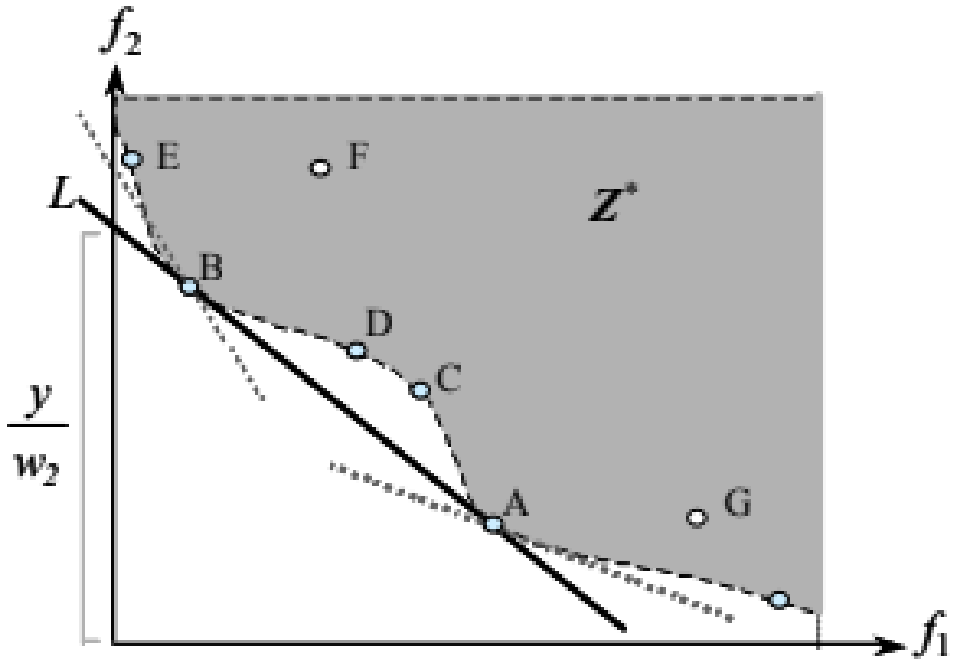
O método usa pesos diferentes para cada objetivo, define-se a função f que é a combinação linear dos objetivos, dada por(ARROYO *et al.*, 2002):

$$\text{Minimizar } f(x) = \sum_{i=1}^{r} w_i(x).f_i(x), \forall x \in X^* \tag{2.8}$$

Sendo, $w_i \geq 0$ o peso que expressa a importância relativa do objetivo f_i frente aos demais. Chankong e Haimes (1983) demonstram que para que uma solução de um problema obtido pelo método da soma ponderada $x*$ seja Pareto-ótima é necessário que dado um vetor de pesos $w = (w1, ..., wr)$, x^* é solução única ou $w_i > 0, \forall i = 1, ..., r$.

Esse método é limitado quando o espaço objetivo é não convexo. Considerando o caso de dois objetivos f_1 e f_2 e seus respectivos pesos w_1 e w_2 para minimizar a função: $y = w_1.f_1(x) + w_2 f_2(x), x \in X^*$. Essa equação pode ser reescrita como $f_2(x) = -\frac{w_1}{w_2} f_1(x) + \frac{y}{w_2}$. A reta descrita por essa equação é uma reta suporte ao espaço objetivo factível Z^* em um ponto Pareto- ótimo. O método da soma ponderada consiste em gerar retas suportes, definidas por valores de w1 e w2, contudo nem todos os pontos Pareto-ótimos admitem retas suportes. Na Figura 2.5, que expressa as equações acima, os pontos C e D não possuem retas suportes, e não podem ser encontrados pela minimização da função $f(x)$ usando o método.

Figura 2.5: Interpretação gráfico do método da soma ponderada.



Fonte: Adaptado de Arroyo *et al.* (2002)

2.3.4.2 Método ε-restrito

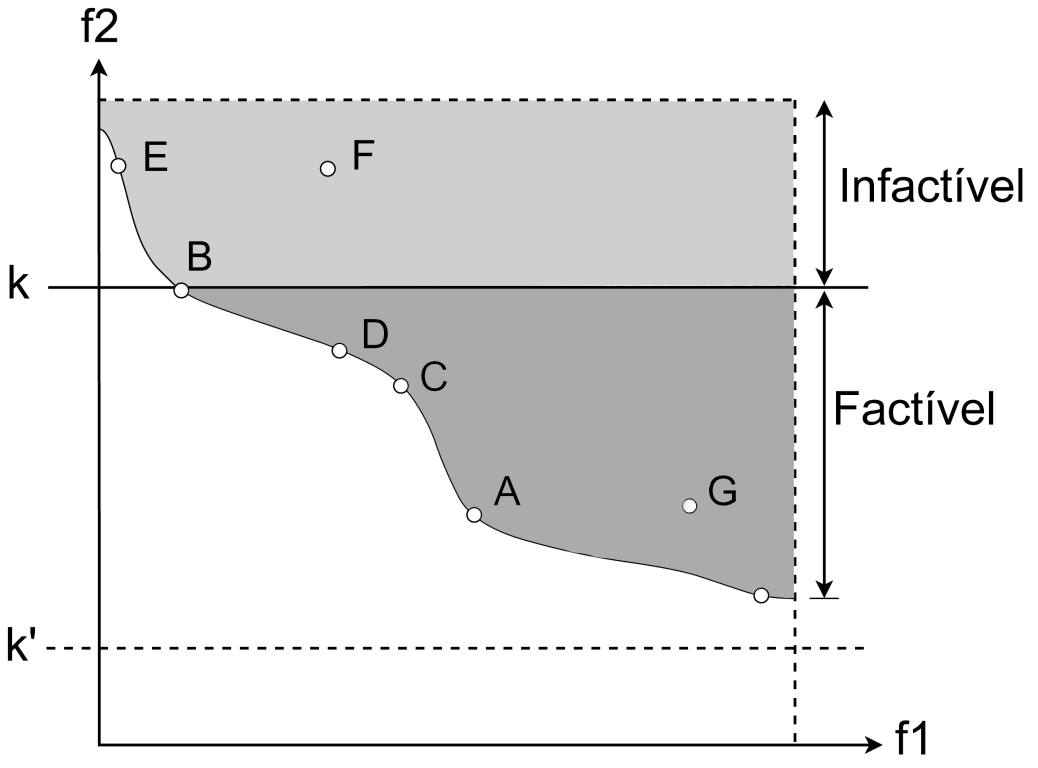
Método ε-restrito é constituído pela minimização do objetivo de maior prioridade sujeito à limitação dos outros objetivos. Adotando f_1 como objetivo prioritário, pode-se formular o método como (ARROYO *et al.*, 2002):

$$\text{Minimizar } f_1(x), \text{ sujeito a } f_i(x) \leq \varepsilon_i,\ i = 2, ..., r,\ \forall x \in X^* \tag{2.9}$$

em que ε_i são limitantes superiores dos objetivos f_i, com $i = 2, ..., r$.

Quando às funções objetivo e as funções de restrição são lineares, tem-se um problema de programação linear. A Figura 2.6, mostra a interpretação gráfica do método para o caso de dois objetivos. A reta $\varepsilon_2 = k$ limita o espaço de soluções. Em alguns casos o limitante superior não é selecionado adequadamente, no caso da figura 2.6, quando $\varepsilon_2 = k'$,tornando o problema sem solução. Cohon (2004) desenvolveu um algoritmo que fornece valores adequados de ε dos limitantes.

Figura 2.6: Interpretação gráfico do método ε-restrito.



Fonte: Adaptado de Arroyo *et al.* (2002)

2.3.5 Métodos de Avaliação de Heurísticas Multiobjetivo

A otimização mono-objetivo avalia uma candidata a solução fazendo a diferença relativa entre os valores da solução heurística e da solução ótima. No entanto, em otimização multiobjetivo o problema apresenta uma maior complexidade. O desafio é mensurar a qualidade de um conjunto avaliado por uma heurística **H** em relação ao conjunto Pareto-ótimo **R**(ARROYO *et al.*, 2002). Serão apresentadas a seguir algumas técnicas de avaliação de heurísticas multiobjetivos.

2.3.5.1 Avaliação Analítica de Daniels

Daniels (1992) desenvolveu um algoritmo que determina a proximidade de um conjunto de pontos heurísticos com o conjunto Pareto-ótimos. A técnica converte cada ponto em valor que depende da importância atribuída ao objetivo e reduz a preferência por um objetivo a medida que o valor do objetivo é incrementado no caso de minimização e decrementado no caso de maximização, assim tem-se:

$$u(z) = \sum_{i=1}^{r} \lambda_i u_i(z_i) \tag{2.10}$$

sendo u_i a função utilidade, que representa a preferência do *i*ésimo objetivo, enquanto o $\sum_{i=1}^{r} \lambda_i = 1$.

Sejam $R = \{z_1^r, ..., z_{Nr}^r\}$ um conjunto com Nr pontos eficientes pertencentes à referência e $H = \{z_1^h, ..., z_{Nh}^h\}$ um conjunto de Nh pontos avaliados pela heurística. Um ponto z_k^r é a melhor alternativa em R se $u(z_k^r) \geq u(z_{k'}^r), \forall k' = 1, ..., Nr$, enquanto z_l^h é a melhor alternativa em H se $u(z_l^h) \geq u(z_{l'}^h), \forall l' = 1, ..., Nh$. O erro relativo de aproximação E é dado por:

$$E = \frac{u(z_k^r) - u(z_l^h)}{u(z_k^r)} \tag{2.11}$$

Os pontos eficientes e heurísticos que são inferiores à função $u(z)$ não são considerados no cálculo do E o que é uma limitante do algoritmo. É, ainda, sugerida uma classe de funções utilidade que produzem preferências que decrescem monotonamente com o incremento de cada objetivo:

$$u_i(z_i) = \left(\frac{\bar{z}_i - z_i}{\underline{z}_i - z_i} \right)^{b_i} \tag{2.12}$$

em que $\bar{z}_i$ e $\underline{z}_i$ são os valores máximo e mínimo do iésimo objetivo dentre os pontos heurísticos e eficientes. É necessário que $b_j > 0$ de modo a se encontrar um valor escalar entre 0 e 1.

2.3.5.2 Medidas de Cardinalidade

Sendo conhecido o conjunto de todos os pontos Pareto-ótimos R, a qualidade pode ser medida pela porcentagem de pontos de referência encontrados pelo método heurístico expresso por $C_1(H) = \frac{|H \cap R|}{|H|} 100\%$. Pode ser impossível obter em tempo razoável um número

significativo de pontos Pareto-ótimos, contudo uma boa aproximação é obtida encontrando pontos aproximados e distribuídos por toda a fronteira Pareto-ótima. Outra medida de cardinalidade é definida como a porcentagem de pontos heurísticos não dominados por pontos de referência $C_1(H) = \frac{|\{z \in H : \nexists z' \in R, z' \text{domina} z|}{|H|} 100\%$. Ambas as medidas não são apropriadas para medir a qualidade de um conjunto aproximado **H**, pois podem ser zero mesmo que H seja uma boa aproximação (ARROYO *et al.*, 2002).

2.3.5.3 Medidas de Distâncias de Czyzak e Jaszkiewicz

Czyzżak e Jaszkiewicz (1998) e Ulungu *et al.* (1998) propõem medidas de distância para medir a proximidade de um conjunto **H** do conjunto de referência **R**. **H** será uma boa aproximação de **R** se fornece informação importante de todas as regiões do conjunto R, ou seja, se a distância entre seus pontos é pequena. Czyzżak e Jaszkiewicz (1998) propuseram as distâncias D_{med}, que é a média das distancias de um ponto $z \in R$ ao ponto mais próximo em **H**, e D_{max} o máximo das distancias mínimas de um ponto $z \in R$ a algum ponto em **H**.

$$D_{med} = \frac{1}{|R|} \sum_{z \in R} min_{z' \in H} d(z', z) \tag{2.13}$$

$$D_{max} = max_{z \in R} \{min_{z' \in H} d(z', z)\} \tag{2.14}$$

Sendo $d = max_{i=1,...,r} \left\{ \frac{1}{\Delta_i} z'_i - z_i \right\}, z' = (z'_1, ..., z'_r) \in H, z = (z_1, ..., z_r) \in R$. Sendo $\Delta_i = maxf_i - minf_i$, sendo $maxf_i = max\{z_i = f_i(x), z = f(x) \in R\}$ e $minf_i = min\{z_i = f_i(x), z = f(x) \in R\}$.

2.3.5.4 Funções de Utilidade de Hansen e Jaszkiewicz

Hansen e Jaszkiewicz (1994), ao considerar que medidas de distância não expressam a preferência do decisor, usualmente representado por funções utilidade, propõem um conjunto de funções utilidade baseadas nas normas ponderadas dadas por:

$$u_p = -\left(\sum_{i=1}^{r} \lambda_i (z_i - z_i^o)^{1/p} \right) p \in \{1, 2, ...\} + \{\infty\} \tag{2.15}$$

Em que $z^o = (z_1^o, ..., z_r^o)$ com $z_j^o = min_{x \in X^*} \{z_i = f_i(x)\}$, $i = 1, ...r$ é ponto ideal e $\lambda_i \geq 0$ são pesos. Caso a preferência do decisor não seja conhecida se define a função utilidade ponderada de Tchebycheff, com $p = \infty$, $u_\infty = -max_i\{\lambda_i(z_i - z_i^o)\}$.

2.3.5.5 Métricas de Van Veldhuizen e Lamont

Veldhuizen e Lamont (2000b) propuseram uma métrica G que é a média das distâncias de um ponto $z' \in H$ ao ponto mais próximo em **R**, mede a proximidade do conjunto **H** ao conjunto **R** em algumas regiões de **R**.

$$G = \frac{1}{|H|}\left(\sum_{z' \in H}(min_{z \in R}\{d(z',z)\})^2\right)^{1/2} \tag{2.16}$$

Em que $d(z',z)$ é a distância Euclidiana entre os pontos z' e z.

Também foi proposta uma métrica da distribuição dos pontos no conjunto **H**: $S = \sqrt{\frac{1}{|H|-1}\sum_{z \in H}(\overline{d} - d_z)^2}$. Em que $d_z = min_{z' \in H}\{\sum_{i=1}^{r}|z_i - z'_i|\}$, sendo $\overline{d} = \frac{1}{|H|}\sum_{z \in H} d_z$.

2.4 Meta-heurísticas multiobjetivo

As meta-heurísticas até então apresentadas, incluindo o GA, prestam-se a resolver problemas com uma única função de avaliação para indicar a melhor solução a ser buscada, no caso do GA a melhor aptidão do indivíduo, no entanto, conforme já abordado, o cenário real poderá exigir que se otimize vários objetivos simultaneamente, ou seja, é comum que se necessite otimizar um problema mais complexo que envolva múltiplos objetivos a serem otimizados simultaneamente. Nesse contexto, surgem diversas técnicas de meta-heurísticas destinadas a resolver problemas multiobjetivos, normalmente derivadas de alguma técnica destinada para problemas mono-objetivos.

Algumas meta-heurísticas especificamente projetadas para resolver problemas com múltiplos objetivos são o multiobjective Differential Evolution Algorithm - MODE (XUE *et al.*, 2003), multiobjective Grey Wolf Optimizer - MOGWO (MIRJALILI *et al.*, 2016), multiobjective Artificial Bee Colony Algorithm - MOABC (HEDAYATZADEH *et al.*, 2010), multiobjective Particle Swarm Optimization Algorithm - MOPSO (COELLO; LECHUGA, 2002) e multiobjective Harmony Search Algorithm - MOHSA (ABEDINIA *et al.*, 2007). Na realidade, muitas abordagens foram desenvolvidas e demonstraram ser eficazes para uma variedade de aplicações lidando com problemas multiobjetivos, uma das mais comuns é a utilização de estratégias evolutivas, com os Algoritmos Evolutivos Multiobjetivos (MOEAs) nos quais se incluem os Algoritmos Genéticos Multiobjetivos (MOGAs).

2.4.1 Algoritmos Evolutivos multiobjetivo (MOEAs)

Uma MOEA visa resolver problemas com a seguinte formulação matemática difundida na literatura (ZITZLER; THIELE, 1999)(SRINIVAS; DEB, 1994)(FONSECA; FLEMING, 1995)(STEUER; STEUER, 1986)(RINGUEST, 2012):

$$\begin{aligned} \text{min/max } \mathbf{y} &= f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_n(\mathbf{x})) \\ \text{Sendo } \mathbf{x} &= (x_1, x_2, ..., x_m) \in X \\ \mathbf{y} &= (y_1, y_2, ..., y_n) \in Y \end{aligned} \tag{2.17}$$

Sendo f uma função vetorial que mapeia uma tupla de m parâmetros (variáveis de decisão) para uma tupla de n objetivos. Em que **x** é denominado vetor de decisão, X é o espaço de parâmetros, **y** é o vetor objetivo e Y é o espaço objetivo. O conjunto de soluções de um problema de otimização multiobjetivo consiste em todos os vetores de decisão para os quais os vetores objetivos correspondentes não podem ser melhorados em nenhuma

dimensão sem degradação em outra (ZITZLER; THIELE, 1999), ou seja, os vetores Pareto ótimos, como já abordado. Em uma MOEA, em regra, o vetor de decisão representará a representação genotípica, sobre o qual serão aplicadas operações evolutivas, visando obter um melhor vetor objetivo.

A distinção entre as MOEAs e as EAs mono objetivo se concentra na dificuldade de otimizar não apenas uma função objetiva mas um vetor objetivo. Nesse contexto, Fonseca e Fleming (1995) categorizam vários MOEAs de acordo com as diferentes estratégias que usam para atribuir aptidão a partir de um vetor objetivo, sendo essas estratégias as abordagens de agregação simples, abordagens não Pareto baseadas em população e abordagens baseadas em Pareto.

Os métodos de agregação combinam os objetivos em uma função escalar que é usada para calcular a adaptabilidade do indivíduo, ou seja, seu desempenho em relação aos objetivos. As abordagens de agregação têm a vantagem de reduzir o vetor objetivo a uma função objetivo escalar, podendo-se, assim, utilizar, com poucas adaptações, técnicas mono-objetivas. Por outro lado, definir a função objetivo dessa maneira requer conhecimento profundo do domínio que muitas vezes não está disponível e não encontram uma família de soluções, como os demais métodos (ZITZLER; THIELE, 1999). Encontram, assim, uma única solução, de acordo com a escolha dos parâmetros definidos para a agregação. Os métodos populares de agregação são a abordagem de soma ponderada, a otimização do vetor alvo e o método de obtenção de metas (ZITZLER; THIELE, 1999)(FONSECA; FLEMING, 1995)(SRINIVAS; DEB, 1994).

Quanto aos algoritmos não-Pareto, buscam encontrar múltiplas soluções não dominadas simultaneamente. Aqueles baseados em população guiam a busca em várias direções ao mesmo tempo, por meio de alteração no critério de seleção durante a fase de reprodução (ZITZLER; THIELE, 1999). A estratégia usualmente utilizada para isso é que cada fração dos indivíduos selecionados para a reprodução são escolhidos de acordo com cada um dos objetivos (FOURMAN, 2014)(KURSAWE, 1992). Outros algoritmos usam combinações lineares dos objetivos em paralelo (HAJELA; LIN, 1992)(ISHIBUCHI; MURATA, 1996).

Quanto às estratégias baseadas em Pareto, a primeira foi proposta por Goldberg (1989). Esses algoritmos usam explicitamente a dominância de Pareto para determinar a probabilidade de reprodução de cada indivíduo. Enquanto os EAs não-Pareto frequentemente não sejam sensíveis à não convexidade dos conjuntos Pareto-ótimos, as técnicas baseadas em Pareto são (ZITZLER; THIELE, 1999)(FONSECA; FLEMING, 1995). É importante salientar, por fim, que muitas técnicas fazem uso de combinações das estratégias de atribuição de aptidão (ZITZLER; THIELE, 1999), por exemplo, em Fonseca e Fleming (1995) e Greenwood *et al.* (1996).

Quando se procura encontrar um conjunto de soluções não dominadas em vez de uma solução de ponto único, procura-se que as soluções encontradas sejam amostras uniformes do conjunto ótimo de Pareto (ZITZLER; THIELE, 1999). Os algoritmos evolutivos simples, que em regra elitismo, tende a convergir para uma única solução, perdendo, normalmente as demais soluções devido a pressão de seleção, ruído de seleção e interrupção do operador (ZITZLER; THIELE, 1999)(MAHFOUD, 1995). Foram criados, assim, alguns métodos para superar esses problemas, ou seja, que visam preservar a diversidade na população, tentam evitar a convergência prematura(ZITZLER; THIELE, 1999). Essas soluções podem ser

agrupadas em técnicas de nichamento e técnicas de não nichamento (MAHFOUD, 1995). As técnicas de nichamento se caracterizam por formar e manter subpopulações estáveis (nichos) enquanto as técnicas de não nichamento usam outras estratégias.

A estratégia de nichamento mais utilizada é o Fitness Sharing (GOLDBERG *et al.*, 1987) e muitos MOEAs implementam o compartilhamento de aptidão (ZITZLER; THIELE, 1999). Essa estratégia se baseia na competição dos recursos disponíveis pelos indivíduos de um determinado nicho, quanto mais indivíduos estiverem localizados na vizinhança de um determinado indivíduo, mais seu valor de aptidão será reduzido. A vizinhança é definida em termos de uma medida de distância $d(i, j)$ especificada pelo raio de nicho σ_{share} (ZITZLER; THIELE, 1999). A função de distância $d(i, j)$ pode operar nos genótipos ou nos fenótipos, ou seja, o compartilhamento pode ocorrer por proximidade de fenótipo, no vetor objetivo, ou de genótipo, nos vetores de decisão.

Entre as técnicas não nichadas, o acasalamento restrito é o mais comum na otimização de função multicritério (ZITZLER; THIELE, 1999). Essa estratégia se baseia na limitação de que para dois indivíduos poderem se acasalar precisam estar dentro de uma certa distância, dada pelo parâmetro σ_{mate}. Com isso, se visa evitar a formação de indivíduos que levam à estagnação em mínimos, buscando melhorar o desempenho global.

Provavelmente as técnicas mais usadas entre as MOEAs são os Algoritmos Genéticos multiobjetivos(MOGAs), até pelo sucesso do GA com um único objetivo. A distinção entre MOEAs que não são MOGAs e os MOGAs as vezes se confundem, de sorte que diversas vezes essas expressões são usadas como sinônimos. Porém, em regra, se usa a mesma diferenciação que se usa para a versão mono-objetivo. As características que costumam ser diferentes em técnicas evolutivas MOGA e não-MOGA é que o MOGA usa uma abordagem baseada na população, em que um conjunto de soluções candidatas (cromossomos) é modificado iterativamente e evolui por meio de um processo de seleção, cruzamento e mutação, tal qual o GA para um único objetivo. Em contraste, os outros MOEAs usam operadores evolutivos, porém sem utilizar como base o GA clássico, podendo se valer, para gerar e avaliar soluções candidatas, de outros algoritmos ou heurísticas especializados. Alguns MOGAs serão abordados na próxima sessão.

2.4.2 Algoritmos Genéticos multiobjetivos(MOGAs)

A seguir, serão apresentados os principais algoritmos genéticos multiobjetivos.

2.4.2.1 Algoritmo Genético Avaliado por Vetor (VEGA)

Proposto por Schaffer (1984), utiliza a estratégia de realizar a seleção para cada objetivo separadamente. Essa seleção é feita repartindo em partes de tamanho igual o grupo de indivíduos para a reprodução, de modo que cada uma dessas partes seja preenchida de acordo com a adaptabilidade para cada um dos objetivos. Os indivíduos são escolhidos pseudo-aleatoriamente da população atual de acordo com o objetivo, de modo que quanto mais adaptado para aquele objetivo maior chance de ser escolhido. Em seguida, o grupo selecionado para o reprodução é embaralhado e são realizados o cruzamento e a mutação.

2.4.2.2 Algoritmo Genético de Hajela e Lin (HLGA)

Proposto por Hajela e Lin (1992), esse algoritmo usa a estratégia de agregação por ponderação de objetivo variável. Trata-se de uma abordagem não-Pareto que usa a soma ponderada dos objetivos para atribuir aptidão (ZITZLER; THIELE, 1999). Os pesos da ponderação somados devem ter valor 1, contudo são variáveis, de modo a buscar múltiplas soluções em paralelo. Para isso, os pesos são codificados no genótipo e a diversidade das combinações de peso se dá pelo compartilhamento de aptidão fenotípica, de tal modo que o algoritmo adapta novas soluções e combinações de peso de forma simultânea. Porém, é necessário aplicar restrições de cruzamento de modo a garantir a estabilidade e agilidade na busca por soluções (ZITZLER; THIELE, 1999).

2.4.2.3 Algoritmo Genético Pareto Nichado (NPGA)

Proposto por Horn *et al.* (1993) e (HORN *et al.*, 1994), esse algoritmo utiliza a dominância de Pareto para realizar a seleção por torneio. Nesta estratégia, durante o torneio, são escolhidos aleatoriamente dois indivíduos concorrentes e um conjunto de comparação formado com outros indivíduos, obedecendo um tamanho de conjunto determinado. Ganha o competidor do torneio que mais tenha dominância de Pareto sobre os membros do conjunto. Caso haja o empate, o indivíduo que tiver menos indivíduos em seu nicho é selecionado para reprodução. Além disso, foi usado o compartilhamento fenotípico nos vetores objetivos.

2.4.2.4 Algoritmo Evolucionário Multiobjetivo Baseado em Decomposição (MOEA/D)

O MOEA/D proposto por Zhang e Li (2007), é um algoritmo evolutivo multiobjetivo baseado em dividir um problema multiobjetivo em subproblemas mono-objetivo e os otimizar simultaneamente. O MOEA/D conta com três estratégias: decomposição, busca de vizinhança e recombinação. Na decomposição o problema é dividido em um conjunto de subproblemas escalares. Já na busca em vizinhança, é realizada a cada geração a troca de soluções entre subproblemas vizinhos, o que visa obter simultaneamente convergência e diversidade das soluções na frente de Pareto. Um subproblema é vizinho quando possui o conjunto de subproblemas mais similares a ele quanto ao desempenho frente aos objetivos. A similaridade pode ser medida usando métricas, como, por exemplo, a distância euclidiana. Em cada geração, uma solução é selecionada de um subproblema com base no critério de seleção, torneio ou roleta. A solução selecionada é comparada com as soluções em sua vizinhança, e a melhor solução é escolhida como ponto de referência para gerar novas soluções no subproblema atual. Já pela recombinação se cria novas soluções candidatas combinando, por cruzamento das soluções de diferentes subproblemas. São então avaliadas usando seus valores de função objetivo e as soluções não dominadas são adicionadas à população.

2.4.2.5 O Algoritmo Evolutivo de Pareto Forte (SPEA)

Proposto por Zitzler e Thiele (1999), utiliza o conceito de densidade no espaço objetivo para manter um conjunto diverso de soluções não dominadas. Para isso, usa uma função

de aptidão que considera tanto a adaptabilidade das soluções candidatas aos objetivos quanto a densidade no espaço objetivo, o que ajuda a manter um conjunto diversificado de soluções. A densidade no espaço objetivo diz respeito à concentração com que as soluções estão distribuídas ao longo da fronteira de Pareto, sendo dada pelo inverso da distância da solução para o ponto da fronteira de Pareto mais próximo, assim, soluções próximas têm uma densidade mais alta. A utilização dessa medida de densidade é manter soluções não-dominadas com baixa densidade para manter a diversidade. O Algoritmo realiza operações de seleção usando uma abordagem de seleção de torneio binário, cruzamento e mutação para gerar uma nova população de soluções candidatas como de costume para as AGs. Outra característica da SPEA é manter as soluções não dominadas encontradas durante o processo de busca, para acompanhar o progresso do algoritmo e garantir que as soluções encontradas sejam diversas e representativas de toda a frente de Pareto. Várias modificações e extensões do SPEA foram propostas, como o SPEA2 (ZITZLER *et al.*, 2001), que aprimora a medida de densidade e o mecanismo de preservação de diversidade.

2.4.2.6 Algoritmo Genético de Ordenação Não Dominada (NSGA)

Proposto por Srinivas e Deb (1994) tem uma abordagem Pareto e se baseia em Goldberg (1989) e Fonseca *et al.* (1993). A atribuição de aptidão se dá com base na ordenação não dominada para classificar os indivíduos em diferentes níveis de não-dominação. Essa ordenação se dá pela identificação dos indivíduos não dominados por nenhum outro indivíduo, classificando-os no primeiro nível de não-dominação. Em seguida, os indivíduos que são dominados apenas pelos indivíduos do primeiro nível são classificados no segundo nível, e assim sucessivamente. A aptidão leva em consideração essa classificação, tendo os indivíduos com níveis mais baixos maior probabilidade de sobreviver à seleção do que os indivíduos nos níveis mais altos. Além disso, para todas as soluções se calcula um operador de diversidade calculado conforme o fenótipo e utiliza-se de função de compartilhamento. Assim, se incentiva a manutenção de uma diversidade genética sem impedir a exploração efetiva no espaço de busca. As variantes mais utilizadas desse algoritmo, que foram utilizadas nesse trabalho, serão abordadas com mais detalhe na próxima seção.

2.4.3 Algoritmos utilizados: NSGA-II e NSGA-III

Por sua ampla utilização na literatura e conhecida simplicidade e eficiência, o NSGA-II e NSGA-III foram as técnicas escolhidas para serem utilizadas nos estudos de caso que serão abordados. A seguir, serão apresentados os dois algoritmos e seus funcionamentos.

2.4.3.1 NSGA-II

O algoritmo NSGA-II (DEB *et al.*, 2002) é um método de otimização multiobjetivo baseado em algoritmos genéticos que visa resolver de forma eficiente problemas com múltiplos objetivos conflitantes, é um aperfeiçoamento da NSGA proposta por Srinivas e Deb (1994). Nesta seção, explicaremos os principais recursos do NSGA-II em detalhes, incluindo a abordagem de classificação não dominada, a técnica de estimativa de distância de aglomeração e os novos operadores de cruzamento e mutação.

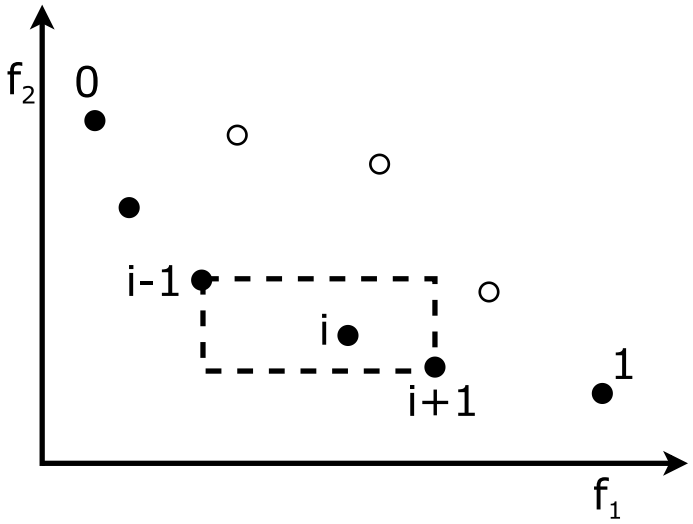
Para identificar eficientemente o conjunto de soluções ótimas de Pareto, o NSGA-II usa a abordagem de classificação não dominada, que classifica a população em diferentes níveis de soluções não dominadas, i_{nivel}, para cada individuo i, abordagem essa herdada da técnica que lhe deu origem, a NSGA. Por essa abordagem, o primeiro nível contém todas as soluções não dominadas, o segundo nível contém todas as soluções dominadas por uma solução não dominada no primeiro nível e assim por diante, até que todas as soluções na população tenham sido atribuídas a um nível. Essa abordagem garante que as soluções ótimas de Pareto sejam preservadas durante todo o processo de busca.

Quanto à estimativa de distância de aglomeração, que visa manter a diversidade na população, o NSGA-II usa, frente ao NSGA, uma nova técnica de estimativa de distância de aglomeração, que mede a distância entre duas soluções na população com base em seus valores objetivos, deixando de usar o método de função de compartilhamento. O NSGA-II passou, ainda, a contar como operador de cruzamento baseado na abordagem de cruzamento binário simulado, que gera novas soluções candidatas combinando informações das soluções pai e com o operador de mutação baseado em mutação polinomial.

Para bem definir o NSGA-II é necessário definir a métrica de estimativa de densidade e o operador de comparação que é empregado no algoritmo para garantir a diversidade das soluções. A estimativa de densidade de soluções em torno de uma solução i pertencente à população é calculada com base na distância ao longo de cada objetivo dos dois pontos a cada lado da solução i ($i-1$ e $i+1$).

O cálculo da distância de aglomeração, é feito para cada uma das funções objetivo sendo depois calculado um valor geral da distância de aglomeração, que representará a densidade de aglomeração. Assim, após a normalização das funções objetivo, se ordena a população de acordo com o seu desempenho na função objetivo. Cada solução recebe, então, um valor de distância igual à diferença normalizada absoluta do desempenho, no objetivo sob análise, das suas duas soluções adjacentes, com exceção das soluções com menor e maior valor naquele objetivo, que estarão nos extremos, sendo atribuído a elas valor de distância infinito. Este cálculo é repetido para cada uma das funções objetivo e obtendo-se o valor geral da distância de aglomeração pela soma dos valores de distância individuais correspondentes a cada objetivo, obtendo-se, para o indivíduo i a distância $i_{distância}$. Assim, utiliza-se como estimativa o perímetro do cubóide representado na Figura 2.7, formado usando-se os vizinhos mais próximos($i-1$ e $i+1$) como vértices, o que é denominado de distância de aglomeração. Este cálculo da distância de aglomeração para dois objetivos pode ser estendido para mais objetivos usando o mesmo procedimento.

Figura 2.7: Representação geométrica do cuboide.



Fonte: Adaptado de Deb et al. (2002)

De posse dessas distâncias, considera-se que uma solução com menor valor está mais aglomerada por outras soluções ao seu redor. Com base nesse raciocínio se define o operador de comparação de aglomeração $\prec_n$ utilizado para a seleção no algoritmo NSGA-II, de modo a orientar a busca em direção a uma frente ótima de Pareto uniformemente espalhada. Tomando dois indivíduos i e j da população, e suas classificações quanto a não dominação (i_{nivel} e j_{nivel}) e a distância de aglomeração ($i_{distância}$ e $j_{distância}$), define-se $\prec_n$ como:

$$i \prec_n j \longleftrightarrow (i_{nivel} < j_{nivel} \text{ OU } \quad ((i_{nivel} = j_{nivel}) \text{ E } \quad (i_{distância} > j_{distância})) \tag{2.18}$$

Assim, dá-se preferência para soluções com o menor nível, ou seja, que tenha mais dominância sobre outras soluções e, caso haja o empate, prefere-se a solução com os arredores menos povoados.

Parte-se da população inicial P_o, que pode ser criada aleatoriamente. Inicialmente, cada solução recebe uma avaliação quanto ao seu nível de não dominância e a população é ordenada quanto a não dominância. Então é realizado o torneio binário para a seleção e posterior cruzamento e mutação, para a criação da população filha Q_o de tamanho N. Para formação das novas populações Q_{i+1}, forma-se uma população combinada $R_i = P_i U Q_i$ de tamanho $2N$. Se procede com a ordenação dos indivíduos de acordo com a não-dominação obtendo-se F, como se une a população atual com a anterior em uma população combinada e se procede o ordenamento, sobrevivendo os melhores indivíduos de R_i se garante a ocorrência do elitismo. Por ser fruto da ordenação da população combinada, F pode ser dividido em $F = F_1 U F_2 U \ldots U F_u$, pertencendo a F_1 as soluções com menor nível de soluções não-dominadas, F_2, as soluções com o segundo menor nível de soluções não-dominadas e assim sucessivamente, até que se tenha as soluções com maior nível de soluções não dominadas F_l.

Definido esse operador pode-se definir o algoritmo NSGA-II, que de forma resumida pode ser definido da seguinte forma:

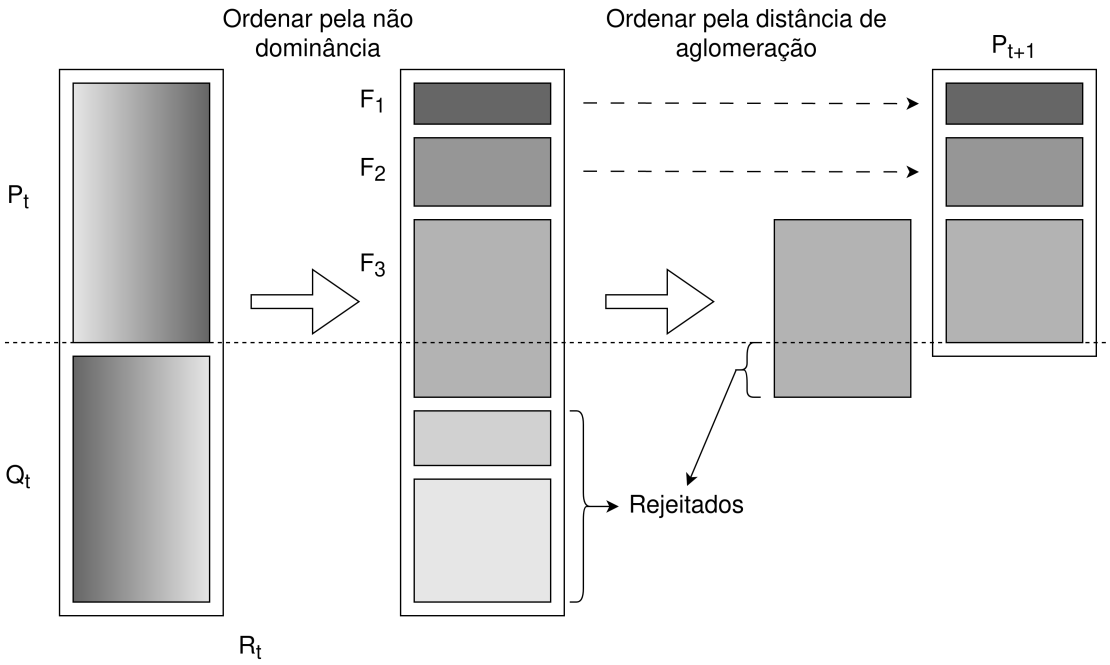
Parte-se da população inicial P_o, que pode ser criada aleatoriamente. Inicialmente, cada solução recebe uma avaliação quanto ao seu nível de não dominância e a população é ordenada quanto a não dominância. Então, é realizado o torneio binário para a seleção e posterior cruzamento e mutação, para a criação da população filha Q_o de tamanho N. Para formação das novas populações P_{i+1}, forma-se uma população combinada $R_i = P_i U Q_i$ de tamanho $2N$.

Se procede com a ordenação dos indivíduos de acordo com a não-dominação obtendo-se F, como se une a população atual com a anterior em uma população combinada e se procede o ordenamento, sobrevivendo os melhores indivíduos de R_i se garante a ocorrência do elitismo. Por ser fruto da ordenação da população combinada, F pode ser dividido em subgrupos, $F = F_1 U F_2 U \ldots U F_l$, representando cada uma das frentes de não dominância, de tal modo que pertence a F_1 as soluções com menor nível de soluções não-dominadas, F_2, as soluções com o segundo menor nível de soluções não-dominadas e assim sucessivamente, até que tenha-se as soluções com maior nível de soluções não dominadas F_l.

Como P_{i+1} também terá tamanho N, deve-se escolher quantos subgrupos F, em ordem crescente de nível, quanto bastem para se obter os N indivíduos, como, em regra, o número de elementos resultantes da união dos i primeiro F_i, não coincidirá com N, devem ser preservados os pertencentes a esses primeiros níveis, complementando com os indivíduos da frente de não-dominância seguinte.

Para escolher entre os membros deste último subgrupo se usa o operador $\prec_n$, escolhendo os menos aglomerados. Os demais indivíduos e as frentes de não-dominância seguintes de F serão rejeitados. A partir desse novo P_{i+1} se realiza a seleção, cruzamento e mutação e se obtém uma nova população Q_{i+1}. Esse procedimento é repetido até que se chegue ao critério de parada. Assim, o operador $\prec_n$ é utilizado, tanto na seleção por torneio quanto durante a fase de redução da população. Embora a distância de aglomeração seja calculada, originalmente, no espaço da função objetivo, ela também pode ser implementada no espaço dos parâmetros (DEB et al., 2002). O procedimento de seleção do NSGA-II pode ser representado pela Figura 2.8.

Figura 2.8: Procedimento de seleção no NSGA-II



Fonte: Adaptado de Deb et al. (2002)

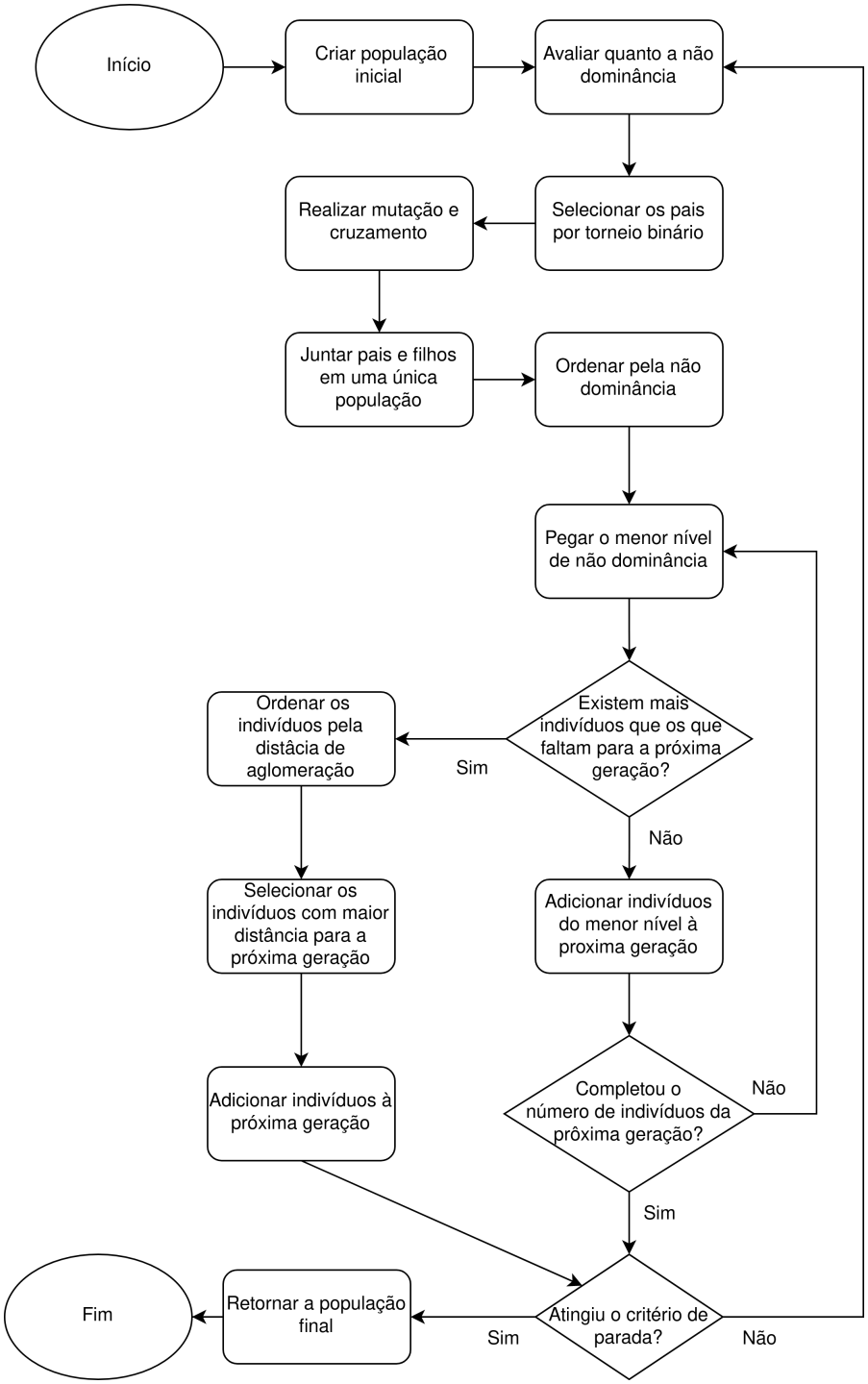
O algoritmo do NSGA-II pode ser expresso pelo fluxograma representado na Figura 2.9.

2.4.3.2 NSGA-III

O NSGA-III é o mesmo do algoritmo NSGA-II com mudanças na etapa da seleção. No NSGA-III, deixou-se de utilizar operador de distância de aglomeração para selecionar soluções da última frente F_l. Passou-se a utilizar um procedimento de seleção com base em nichamento, partindo da determinação de pontos de referência em um hiperplano, da normalização adaptativa da população e associação das soluções candidatas aos pontos de referência(Z), formando nichos. Para a seleção da população, se inicia utilizando um procedimento semelhante ao NSGA-II. Parte-se da população inicial P_o, realiza-se o torneio binário para a seleção e posterior cruzamento e mutação, para a criação da população filha Q_o. Para formação das novas populações P_{t+1}, forma-se uma população combinada $R_t = P_t U Q_t$ e se procede com a ordenação dos indivíduos de acordo com a não-dominação obtendo-se F, como no NSGA-II.

Os indivíduos de R_t são adicionados a população S_t até que $|S_t| = |P_{t+1}|$, quando se formará a nova população atribuindo-se os indivíduos de S_t a P_{t+1}. Como no NSGA-II, caso exista um l tal que, a soma do número de membros das populações dos níveis de frente não dominada, do 1 ao l, seja igual a N, apenas se inclui todos os membros na nova população sem nenhuma outra operação ser necessária. Caso isso não seja possível, escolhe-se l tal que $N = |U_{i=1}^{l-1} F_i| + K$, sendo K o número de membros pertencentes a F_l necessários para completar P_{t+1}. Então, os membros de F_1 a F_{l-1} são adicionados a nova população e deve-se escolher K membros de F_l para também serem adicionados. No

Figura 2.9: Fluxograma de funcionamento de um NSGA-II.



Fonte: Autoria própria

NSGA-III, a escolha desses K membros de F_l não são mais selecionados por $\prec_n$ mas por um procedimento de nichamento. O algoritmo do NSGA-III pode ser expresso pelo fluxograma representado na Figura 2.10.

Para fazer o nichamento, garantindo a diversidade nas soluções obtidas, o NSGA-III usa os pontos de referência para definir um hiperplano normalizado, que é igualmente inclinado a todos os eixos objetivo e intercepta cada eixo no um. Os pontos de referência (Z) iniciais podem ser fornecidos pelo usuário ou definidos de forma estruturada usando a abordagem de Das e Dennis (1998). Os pontos de referência devem ser definidos de modo a ficarem bem distribuídos pelo hiperplano normalizado, de modo que as soluções obtidas também sejam amplamente distribuídas na frente ótima de Pareto ou próximo a ela. Atendendo a essa premissa, é provável que o algoritmo encontre soluções próximas do ótimo de Pareto correspondentes aos pontos de referência fornecidos (DEB; JAIN, 2014). Assim, além de se preservar as soluções não dominadas, o algoritmo preserva soluções associadas a cada ponto de referência, para manter a distribuição das soluções ao longo da frente de Pareto.

Para normalização do hiperplano e dos membros da população nesse hiperplano, primeiramente, se define o ponto ideal da população pela identificação do valor mínimo (z_i^{min}) para cada função objetivo $i = 1,2,...,M$, obtendo-se o ponto ideal $\bar{z} = (z_1^{min}, z_2^{min}, ..., z_M^{min})$. Cada eixo objetivo do hiperplano f_i é então subtraído do z_i^{min} de modo que o ponto ideal se torne um vetor zero. As funções objetivo normalizadas são dadas por $f_i'(x) = f_i(x) - z_i^{min}$. Define-se, ainda, os pontos extremos $(z^{i,max})$ para cada objetivo i, que torna a função escalar de realização correspondente a cada $f_i'(x)$ mínimo, de modo que pode-se obter um a_i que dá o valor escalar da origem a $(z^{i,max})$ para cada $f_i'(x)$ equivalente. As funções objetivo podem então ser normalizadas como:

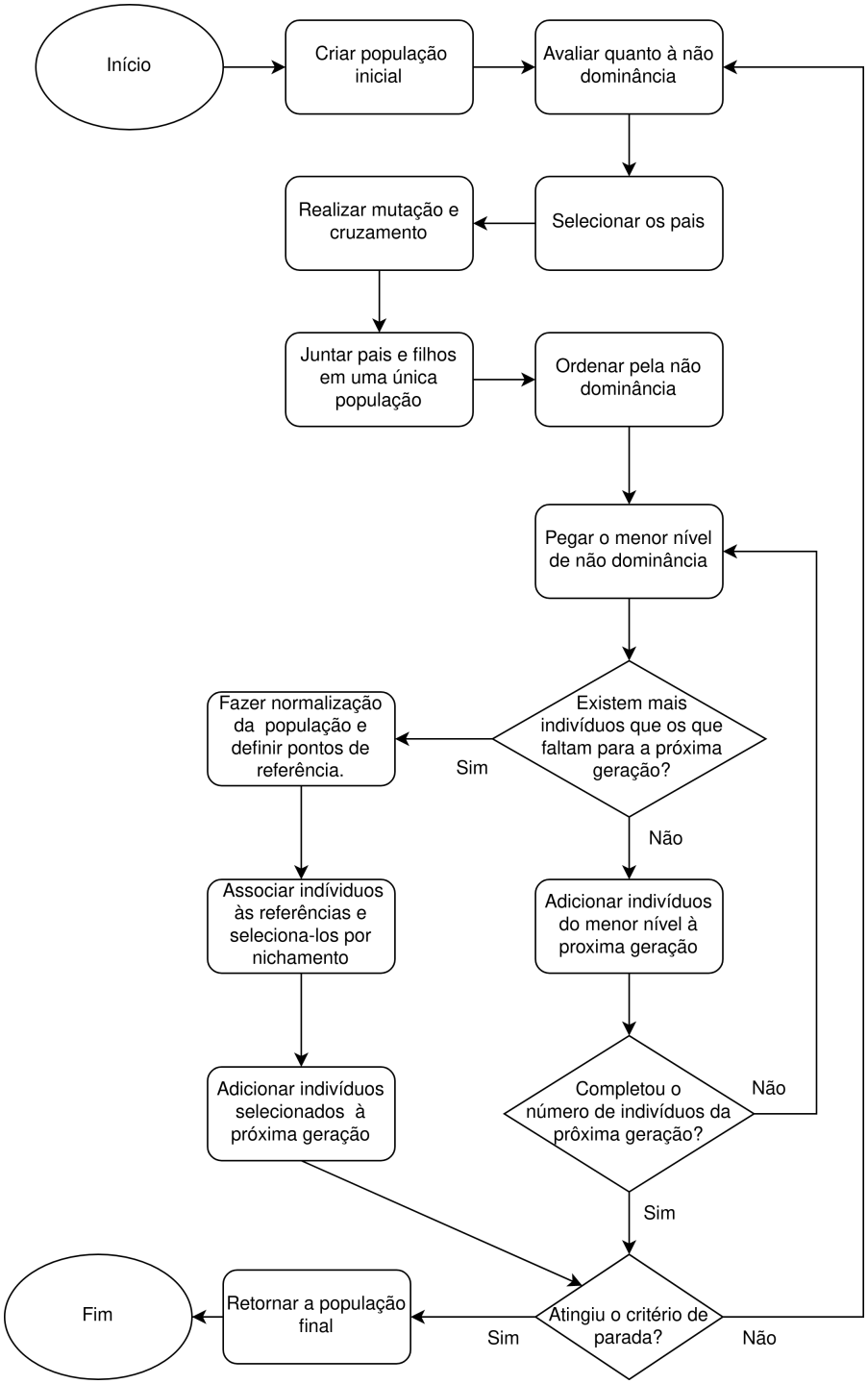
$$f_i^n(x) = f_i'(x)/a_i, \text{ para } i = 1,2,...,M. \tag{2.19}$$

Assim, cada eixo objetivo tornou-se normalizado $f_i^n = 1$, e o hiperplano normalizado definido por esses pontos de interseção terá $\sum_{i=1}^{M} f_i^n = 1$. Caso o Z tenha sido fornecido pelo usuário, obtém-se, ainda, aplicando o procedimento expresso pela Equação 2.19, Z normalizado, denominado de Z^r, caso os pontos já tenham sido definidos pela abordagem de Das e Dennis (1998) eles já estão normalizados e portando $Z^r = Z$. O procedimento de normalização e criação do hiperplano é feito a cada geração usando pontos extremos já encontrados desde o início da execução do algoritmo.

Após a normalização, associa-se cada membro da população a um ponto de referência. Para isso, define-se uma linha de referência para cada ponto de referência, unindo-o com a origem. Em seguida, calculamos a distância perpendicular de cada membro normalizado da população de S_t a cada uma das linhas de referência. O ponto de referência cuja linha de referência está mais próxima de um membro da população no espaço objetivo normalizado é considerado associado ao membro da população. A partir disso, é possível fazer uma contagem de nicho ρ_j, que é o número de membros da população associados a um dado ponto de referência j. Permitindo o nichamento partindo-se dos pontos de referência.

Deve-se finalmente utilizar a estratégia de preservação de nichos para escolha da população. Primeiro, identifica-se o ponto de referência $\bar{j}$ com ρ mínimo, caso haja empate um ponto é escolhido aleatoriamente. Se $\rho_{\bar{j}} \geq 1$, adiciona-se a P_{t+1}, se existir um

Figura 2.10: Fluxograma de funcionamento de um NSGA-III.



Fonte: Autoria própria

membro da frente F_l que está associado ao ponto de referência $\bar{j}$, caso haja mais de uma solução membro de F_l, será escolhido um aleatoriamente ou levando em consideração outro critério de preservação da diversidade e a contagem de $\rho_{\bar{j}}$ é incrementada em um. Contudo, se $\rho_{\bar{j}} = 0$, não havendo soluções em P_{t+1} associado ao ponto de referência $\bar{j}$, quando existir um ou mais membros na frente de não-dominância F_l que estão associadas ao ponto de referência $\bar{j}$, deve-se adicionar a P_{t+1} aquela solução que tiver a menor distância perpendicular à linha de referência de $\bar{j}$ e incrementar em 1 a $\rho_{\bar{j}}$. Quando, porém, não há elementos na frente F_l associados ao ponto de referência $\bar{j}$, não se deve levar em consideração o ponto de referência na corrente geração. Esse procedimento de preservação de nicho é repetido até completar P_{t+1}. O procedimento de atribuir um indivíduo a um ponto de referência e de fazer a seleção por nichamento pode ser observado no pseudocódigo apresentado no Algoritmo 1.

Algoritmo 1: Nichamento

Dados: Z^r, S_t
Saída: P_{t+1}

para cada $z \in Z^r$ **faça**
  Calcule a linha de referência $w = z$;
fim
para cada $s \in S_t$ **faça**
  para cada $w \in Z^r$ **faça**
    Calcular $d^{\perp}(s,w) = \| (s - w^T sw / \| w \|^2) \|$;
  fim
  Atribuir $\pi(s) = w : argmin_{w \in Z^r} d^{\perp}(s,w)$;
  Atribuir $d(s) = d^{\perp}(s, \pi(s))$;
fim
$k = 1$;
enquanto $k \leq K$ **faça**
  $J_{min} = \{ j : argmin_{j \in Z^r} \rho_j \}$;
  $\bar{j} = ElementoAletorio(J_{min})$;
  $I_{\bar{j}} = \{ s : \pi(s) = \bar{j}, s \in F_l \}$;
  se $I_{\bar{j}} = \emptyset$ **então**
    se $\rho_{\bar{j}} = 0$ **então**
      $P_{t+1} = P_{t+1} \cup \left(s : argmin_{s \in I_{\bar{j}}} d(s) \right)$;
    senão
      $P_{t+1} = P_{t+1} \cup ElementoAletorio(I_{\bar{j}})$;
    fim
    $\rho_{\bar{j}} = \rho_{\bar{j}} + 1$;
    $F_l = F_l \setminus s$;
    $k = k + 1$;
  senão
    $Z^r = Z^r / \{ \bar{j} \}$;
  fim
fim

2.5 Avaliação de Controladores

Os índices de avaliação servem para estabelecer um critério principal para avaliar o desempenho de controladores associados as respectivas funções *fitness*, ou funções de avaliação, possibilitando avaliar e comparar os diferentes controladores projetados para uma determinada planta ou processo. O índice deve indicar corretamente se uma solução é melhor que a outra, permitindo identificar qual resposta está mais próxima da resposta ótima. Vários índices são encontrados na literatura para esse fim, os mais comuns são os que envolvem a integração do erro em relação ao *setpoint*, podem ser citados como exemplos: Integral do Erro Absoluto (IEA), Integral do erro absoluto ponderado no tempo

(IEAT), Integral do Erro Quadrático (ISE).

2.5.1 Integral do Erro Absoluto (IEA):

É dada pela equação 2.20. Do inglês, *Integral Absolute Error* (IEA).

$$IEA = \frac{1}{L}\sum_{n=1}^{L}|e(n)| \tag{2.20}$$

em que $e(n)$ representa o erro e L representa o número total de amostras, sendo n o instante de tempo.

2.5.2 Integral do Erro Absoluto Ponderado no Tempo (IEAT):

É dada pela equação 2.21. Do inglês, *Integral Time-weighted Absolute Error* (IEAT).

$$IEAT = \frac{1}{L}\sum_{n=1}^{L}n|e(n)|. \tag{2.21}$$

em que $e(n)$ representa o erro, n expressa o instante de tempo e L representa o número total de amostras.

2.5.3 Integral do Erro Quadrático (ISE):

É dada pela equação 2.22. Do inglês *Integral Squared Error* (ISE).

$$ISE = \frac{1}{N}\sum_{n=1}^{L}e(n)^2 \tag{2.22}$$

em que $e(n)$ representa o erro e L representa o número total de amostras, sendo n o instante de tempo.

2.6 Transformada *Wavelet*

Fourier, no século XIX, mostrou que qualquer função periódica pode ser expressa como uma soma infinita de funções exponenciais complexas periódicas. Anos depois, suas teorias foram generalizadas para funções não periódicas (STRANG; NGUYEN, 1996).

Em 1965, desenvolveu-se um novo algoritmo denominado de Transformada Rápida de Fourier, tornando a transformada de Fourier ainda mais popular (STRANG; NGUYEN, 1996). A transformada de Fourier fornece informações de frequência e fase de um dado sinal, apenas indicando se um certo componente de frequência existe ou não (STRANG; NGUYEN, 1996). Essas informações são independentes de quando o componente aparece no tempo, o que não é adequado para algumas aplicações. Assim sendo, foi desenvolvida

uma representação de frequência de tempo linear chamada *Short Time Fourier Transform* (STFT).

Na STFT, o sinal é dividido em segmentos suficientemente pequenos, ao ponto de poder-se assumir que o sinal é estacionário, escolhendo para isso uma função janela. Contudo, a STFT enfrenta o desafio do Princípio de Incerteza de Heisenberg (MANJUNATH; MA, 1996), o que originalmente se aplicaria à medição da velocidade e localização de partículas em movimento, pode ser aplicado, nesse caso, à informação de frequência e de tempo de um sinal. Por esse princípio, não é possível conhecer a representação exata de frequência-tempo de um sinal simultaneamente, não sendo possível saber quais componentes espectrais existem em um dado instante de tempo. Em vista disso, quanto mais estreita a janela escolhida, melhor será a resolução do tempo, se aproximando mais de um sinal estacionário. Contudo, há a progressiva perda de resolução de frequência, assim, dependendo da aplicação, deve-se escolher uma janela adequada.

Quando os componentes de frequência estão bem separados, é possível com pouco sacrifício na frequência, obter uma boa resolução no tempo. No entanto, se este não for o caso, torna-se difícil encontrar uma boa janela (WANG *et al.*, 1997)(ABRAMOWITZ *et al.*, 1972)(CERVENKA; MOUSTIER, 1993). Uma abordagem alternativa seria analisar o sinal usando a transformada *wavelet*, que permite a análise do sinal em diferentes frequências com diferentes resoluções. Assim, pode-se decompor e descrever um sinal em funções *wavelet*, podendo dessa forma representá-lo em diferentes escalas de frequência e de tempo, representando uma ferramenta poderosa para a análise de sinais (MAGALHÃES, 2007)(VILANI; SANCHES, 2013). A análise *wavelet* possui aplicações em compressão de dados, separação de componentes, eliminação de ruído e detecção de autossemelhança.

A transformada *wavelet* é projetada para dar uma alta resolução de tempo e baixa resolução de frequência para altas frequências e alta resolução de frequência e baixa resolução de tempo para baixas frequências, o que faz sentido, especialmente quando o sinal tem componentes de alta frequência com durações curtas e os componentes de baixa frequência têm longa duração. Existem duas diferenças principais entre a Transformada de Fourier Curta (STFT) e a Transformada *Wavelet* Contínua (DARILMAZ, 2006):
1. Não se toma as transformadas de Fourier dos sinais obtidos do janelamento (LAINE; FAN, 1993)(STRANG; NGUYEN, 1996).
2. A largura da janela é alterada conforme a transformada é computada para cada componente espectral, o que é provavelmente a característica mais significativa da transformada *wavelet*.

A *wavelet* mãe é a função *wavelet* usada como protótipo para gerar as outras funções usadas como as janelas da transformada *wavelet*, pela sua mudança de escala, por compressão ou expansão; e translação, que é a localização da janela e como a janela é deslocada através do sinal no tempo. A relação entre escala e frequência é que as escalas baixas correspondem a altas frequências e escalas altas a baixas frequências.

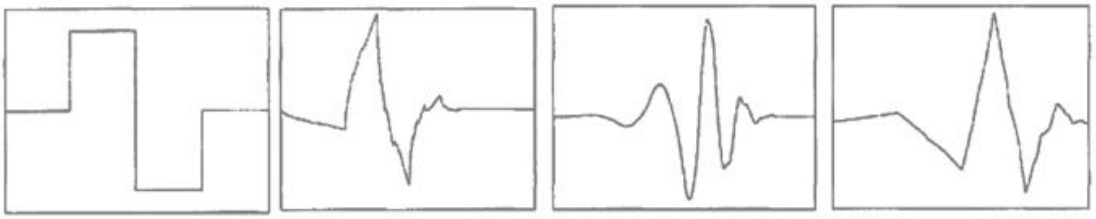
Para ser considerada uma *wavelet*, uma função, $\psi(t)$ tem que atender as seguintes características:

1- A área total sob a curva da função é 0: $\int_{-\infty}^{\infty} \psi(t)dt = 0$.

2- A energia da função é finita: $\int_{-\infty}^{\infty} |\psi(t)|^2 dt < L$, com $L \in \mathbb{R}_+^*$.

Alguns exemplos de funções *wavelets* são encontrados na Figura 2.11.

Figura 2.11: Algumas funções *wavelet*.



Fonte: Darilmaz (2006)

A transformada *wavelet* contínua de um sinal, $f(t)$, é dada por (DARILMAZ, 2006):

$$C(a,b) = \int_{-\infty}^{\infty} f(t)\Psi^*_{a,b}(t)dt \tag{2.23}$$

em que, $\Psi^*_{a,b}(t)$ é o conjugado de $\Psi_{a,b}(t)$ dado por:

$$\Psi_{a,b}(t) = a^{-\frac{1}{2}}\Psi\left(\frac{t-b}{a}\right). \tag{2.24}$$

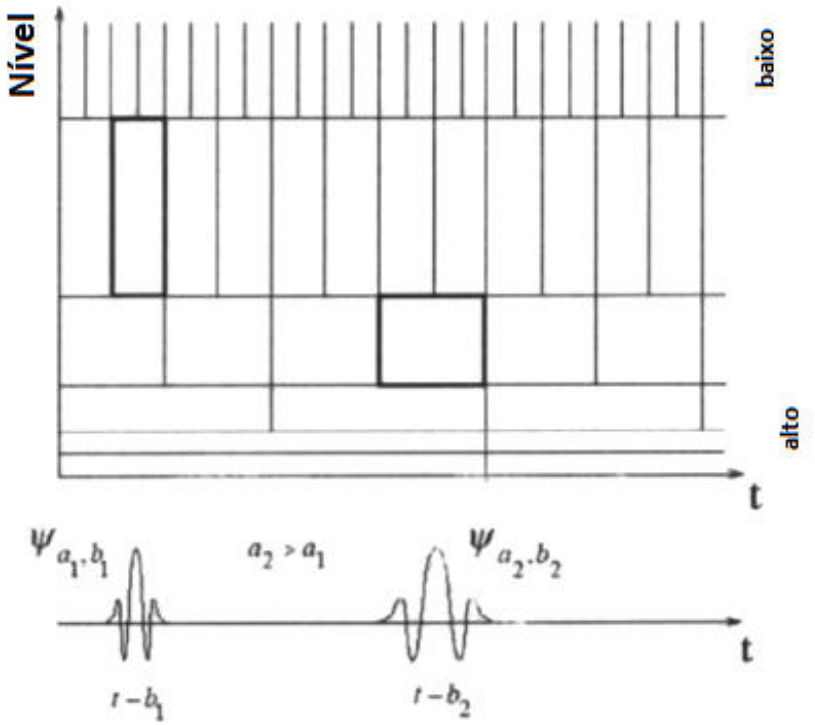
A Equação 2.24 apresenta $\Psi_{a,b}(t)$, que é uma função de janela da *wavelet* mãe, sendo a a escala e b a translação. Assim, $\Psi_{a,b}(t)$ é obtida pelas operações de translação e escala da função *wavelet* mãe $\psi(t)$. Na Figura 2.12, podem-se observar estas duas operações, a translação desloca a janela no tempo, enquanto a escala diminui ou aumenta a duração do sinal. É a operação de escala que permitirá guardar a informação da frequência do sinal, quanto mais curto, ou seja, quanto menor a escala maior será a frequência correspondente. A partir das equações anteriores, pode-se notar que, ao dilatar uma *wavelet* no tempo, diminui-se seu detalhamento no domínio da frequência. Além de diminuir o detalhamento de frequência, a frequência central da *wavelet* muda para frequências mais baixas.

Isso descreve a transformada *wavelet* contínua como uma filtragem de passagem de banda do sinal de entrada. Os coeficientes da transformada *wavelet* contínua em escalas mais baixas representam energia no sinal de entrada em frequências mais altas, enquanto os coeficientes da transformada *wavelet* contínua em escalas mais altas representam energia no sinal de entrada em frequências mais baixas. No entanto, diferentemente da filtragem de passagem de banda de Fourier, a largura do filtro de passagem de banda na transformada *wavelet* contínua é inversamente proporcional à escala. Isso decorre das relações de incerteza entre o detalhamento de tempo e frequência de um sinal: quanto mais amplo o detalhamento de um sinal no tempo, mais estreito será o detalhamento de frequência. O relacionamento inverso também se mantém.

Pode-se também interpretar a transformada *wavelet* contínua (CWT) como uma filtragem do sinal baseada em frequência, sendo possível reescrevê-la como uma transformação de Fourier inversa, conforme a Equação 2.25.

$$C(a,b) = \int_{-\infty}^{\infty} \hat{f}(\omega)\hat{\Psi}^*(a\omega)e^{i\omega b}d\omega \tag{2.25}$$

Figura 2.12: As operações de escala e translação na função *Wavelet* mãe



Fonte: Adaptado de Darilmaz (2006)

sendo $\hat{f}(\omega)$ e $\hat{\Psi}(\omega)$ as transformadas de Fourier do sinal e da *wavelet*.

A Transformada Wavelet conforme a Equação 2.25 é uma transformada de Fourier inversa de um produto das transformadas de Fourier. Existem algoritmos eficientes para o cálculo da transformada discreta de Fourier e sua inversa, possibilitando menor gasto computacional com o uso do algoritmo fft. Para isso tem-se:

$$\tilde{\Psi}_a(t) = \frac{1}{a}\Psi^*\left(\frac{-t}{a}\right) \tag{2.26}$$

Assim, pode-se expressar a transformada *wavelet* como:

$$C(a,b) = \int_{-\infty}^{\infty} f(t)\tilde{\Psi}_a(b-t)dt = (f * \tilde{\Psi}_a)(b) \tag{2.27}$$

A versão discretizada, supondo que a sequência de entrada seja um vetor $x[n]$ de comprimento N, será:

$$W_a[b] = \sum_{n=0}^{N-1} x[n]\tilde{\Psi}_a[b-n] \tag{2.28}$$

Capítulo 3

Metodologia

A otimização de controladores, no geral, se baseia na utilização de heurísticas mono-objetivos seguindo um único índice baseado em índices de avaliação de controladores, usualmente baseados em erro, ou a soma ponderada desses índices. Contudo, IEA e ISE, não apresentam grande diferença de comportamento nas respostas encontradas, sendo muitas vezes comutáveis. Já o IEAT visa privilegiar o bom comportamento no regime permanente com a ponderação no tempo, desse modo, carrega a mesma informação contida no IEA acrescida da informação temporal. Assim, há pouco acréscimo de informação na utilização da ponderação desses índices como função custo. Essa abordagem é adotada em Issa (2023), Saxena *et al.* (2023) e Ozumcan *et al.* (2023) e a comparação da eficiência desses índices é realizada em Gupta *et al.* (2023).

Outra estratégia é utilizar a soma ponderada de característica da resposta ao degrau do sistema como o tempo de subida, tempo de estabilização, *overshoot* e etc. Essa estratégia a princípio é mais promissora por refletir princípios que um sintonizador humano levaria em consideração ao avaliar e ajustar os controladores, procurando construir um controlador que da melhor forma possível atenda a esses requisitos simultaneamente. Contudo essa estratégia apresenta limitações, pois não são definidos para sinais de referência mais complexos que um degrau, faltando generalidade.

Visando resolver essa limitação, buscando uma estratégia mais genérica, nesse trabalho, é proposto o uso da decomposição e análise do erro por meio de transformada *wavelet*, analisando nos níveis adequados os descritores do erro que reflitam o comportamento da resposta da planta controlada, em relação aos aspectos anteriores e outros como a rejeição a ruído e perturbação. A ideia central da análise em *wavelet* consiste em decompor um sinal a diferentes níveis de resolução, processo conhecido como multirresolução (VILANI; SANCHES, 2013), ou multinível. A análise multinível permite cobrir um amplo espectro de frequências, porém, sem perder a noção do comportamento temporal. A transformada *wavelet* permite avaliar a mudança de frequências de um sinal ao longo do tempo, sendo seus descritores usados para aferir a similaridade de sinais das mais diversas naturezas.

Neste contexto, Kukharchuk *et al.* (2017) usaram *wavelet* para análise espectral de processos de vibração em unidades hidrelétricas, Pant *et al.* (2021) aplicou *wavelet* para a detecção de falhas de energia em transformadores, Ke (2022) empregou análise *wavelet* para detecção de danos em estruturas de cabos de pontes de longo vão e Zarachoff *et al.* (2021) propuseram uma estratégia de reconhecimento auditivo multibanda por análise de componentes principais *wavelet* e Sornsen *et al.* (2021) usaram transformada *wavelet*

para detecção de sinal de descarga parcial em geradores. No geral, a estratégia utilizada nesses trabalhos consiste em obter a transformada *wavelet* do erro e escolher determinados descritores, comparando determinados coeficientes *wavelet* das medições com os de uma dada referência, por meio de distância euclideana ou outras estratégias de medição.

Nas próximas seções, será tratada a estratégia para a otimização no domínio *Wavelet* e a metodologia para realização dos estudos de caso para avaliar a eficácia da abordagem proposta.

3.1 Otimização no domínio *Wavelet*

Diante do exposto, será adotada estratégia similar à usada para a análise e processamento de sinais com outras finalidades, para verificar a similaridade entre a resposta de um sistema de controle e o *setpoint* desejado. A formulação do problema quanto ao domínio *wavelet* para a otimização se baseia na análise em diferentes faixas de frequência, o que abre uma maior gama de possibilidades, pois, muitas vezes, se conhece o espectro de frequência de aspectos relevantes do sistema como do ruído, das perturbações, ou mesmo do regime permanente e do transitório e, a partir dessa decomposição, pode-se definir o desempenho desejado para um dado controlador. Assim, a partir da identificação de faixas de frequência é feita a escolha de objetivos a serem otimizados.

Assim, se expande um problema que seria a princípio mono-objetivo, visando apenas minimizar o erro, em um problema multiobjetivo, reduzindo os descritores de erro referente a cada aspecto da resposta do sistema. Como usualmente esses aspectos a serem otimizados podem ser conflitantes, se buscará a utilização de estratégias multiobjetivo para se encontrar a solução mais adequada.

Em resumo, a metodologia proposta consiste na formulação do problema usando descritores *wavelet*, sendo feita a otimização de índices baseados nesses descritores, utilizando os índices obtidos como funções custo a cada iteração do algoritmo de otimização multiobjetivo. Desse modo, pode-se dividir a estratégia proposta em duas etapas: a etapa de projeto e a etapa de otimização. Na etapa de projeto, são definidas as faixas de frequência a serem utilizadas e na etapa de otimização, é aplicado um algoritmo multiobjetivo sobre índices de avaliação baseados em *wavelet*.

3.1.1 Etapa de projeto

A definição das faixas de frequência a serem utilizadas leva em consideração que as implementações de otimização são feitas por meio de computadores digitais, assim, pode-se repartir a faixa de frequência de 0 até $N/2$, devido ao teorema de Nyquist, sendo N a frequência de amostragem. O conhecimento de cada faixa de frequências pode ser obtido de forma teórica ou heurística.

É possível conhecer o espectro de frequência da referência e da resposta do sistema de forma teórica a partir do modelo aproximado da planta. De posse do modelo, pode-se conhecer as frequências de corte, ou os polos e zeros aproximados do sistema. Além disso, pode-se ter o modelo matemático de aspectos como ruídos, perturbações, entre outros. De posse desses modelos defini-se as faixa de frequência em que esses aspectos irão ocorrer.

De forma experimental, pode-se definir as frequências, a partir da resposta do sistema com o controlador que será usado como ponto de partida para a otimização. De posse da resposta encontrada, pode-se estimar as frequências apresentadas com base na medida aproximada dos períodos das oscilações presentes, estimando as faixas de frequência apresentadas, associando-as a cada aspecto sobre analise, com auxilio de conhecimento de especialista sobre o sistema ou conhecimento teórico parcial, caso trate-se de um sistema caixa cinza. Assim, é possível escalonar as faixas de frequência em que cada um desses aspectos deverá ocorrer e aplicar a otimização.

3.1.2 Etapa de otimização

Para a analise proposta, que avalia faixas de frequências e não uma frequência específica, partindo das Equações 2.27 e 2.28 pode-se fazer:

$$W_{a_1a_2}[b] = \sum_{i=a_1}^{a_2} \sum_{n=0}^{N-1} x[n]\tilde{\Psi}_i[b-n] \tag{3.1}$$

Definindo:

$$\tilde{\Psi}_{a_1a_2}[n] = \sum_{i=a_1}^{a_2} \tilde{\Psi}_i[n] \tag{3.2}$$

Temos,

$$W_{a_1a_2}[b] = \sum_{n=0}^{N-1} x[n]\tilde{\Psi}_{a_1a_2}[b-n] \tag{3.3}$$

Sendo a_1 e a_2 a frequência mínima e máxima a serem consideradas na análise, respectivamente. Ressalva-se que a máxima frequência analisada é limitada pelo teorema de Nyquist.

Objetiva-se obter descritores para cada faixa de frequência definida na etapa de projeto, sobre a resposta do sistema com o controlador sob análise y e sinal de referência r. A comparação de similaridade é feita pela diferença absoluta entre os descritores de r e de y, nos respectivos correspondentes em frequência. Assim, pode-se definir índices baseados em decomposição *Wavelet* para cada faixa de frequência sob análise, IDW, pelo somatório da diferença absoluta entre os descritores para cada faixa de frequência, em b de 0 a $N-1$. Assim, partindo da Equação 3.3, tem-se:

$$IDW_{a_1a_2} = \sum_{b=0}^{N-1} \left| \sum_{n=0}^{N-1} r[n]\tilde{\Psi}_{a_1a_2}[b-n] - \sum_{n=0}^{N-1} y[n]\tilde{\Psi}_{a_1a_2}[b-n] \right| \tag{3.4}$$

Sendo $e[n]$ o erro, tem-se:

$$IDW_{a_1a_2} = \sum_{b=0}^{N-1} \left| \sum_{n=0}^{N-1} e[n]\tilde{\Psi}_{a_1a_2}[b-n] \right| \tag{3.5}$$

Desse modo, serão utilizados os $IDW_{a_1a_2}$ para cada faixa de frequência desejada como

objetivo a ser otimizado por uma estratégia multiobjetivo.

3.2 Metodologia para realização dos estudos de caso

Para a avaliação da metodologia proposta, será aplicada a estratégia de otimização apresentada para quatro sistemas. Para cada um dos sistemas de controle se estabeleceu faixas de frequência a serem otimizadas, definindo os índices *IDW* a serem otimizados por algoritmos de otimização. Para a otimização dos *IDW* será adotado NSGA-II e NSGA-III como técnicas de otimização multiobjetivo, pelo seu amplo uso na literatura, simplicidade e conhecida eficiência. As implementações do NSGA-II e do NSGA-III para Matlab utilizadas foram as trazidas em Heris (2015) e Heris (2016), respectivamente.

Escolheu-se a utilização de controladores da família Fuzzy-PID do tipo sugeno, para todos os estudos de caso abordados, partindo de um controlador já conhecido para cada sistema, sendo aplicada um variação pseudorrandômica sobre ele, para originar diferentes parâmetros referentes às funções saída dos controladores *fuzzy* Sugeno, o processo foi repetido até formar uma população inicial de soluções candidatas para cada estudo de caso, de tal modo que todos os algoritmos partam da mesma população inicial para um mesmo sistema.

Os parâmetros de cada controlador, funções de pertinência de entrada e Sugeno de saída, serão codificados como *arrays* de valores, representando cromossomos, e se aplicará os algoritmos para sua otimização em função dos $IDWs$ escolhidos para cada sistema. Para realização da avaliação da otimização, obtendo os $IDWs$, os sistemas controlados serão simulados na plataforma Simulink do Matlab, com o *tollbox Fuzzy* para a implementação dos controladores.

Nas técnicas de otimização multiobjetivo adotadas, os problemas serão selecionados pela não dominância, de tal modo que se obterá uma população de soluções otimizadas para todos os objetivos simultaneamente, devendo o projetista posteriormente escolher qual indivíduo será o mais adequado para resolver o problema. Para comparação realizar-se-á a otimização dos mesmos sistemas de controle utilizando o GA com o IEA e o IEAT como função custo. Serão fixados heuristicamente a quantidades de indivíduos para a população e o número de gerações, de modo a permitir a comparação da eficiência das metodologias de acordo com o caso de estudo.

No próximo capítulo, serão apresentados diferentes estudos de casos que utilizaram a abordagem aqui apresentada para otimizar controladores *fuzzy* para dados sistemas escolhidos, com os respectivos resultados.

Capítulo 4

Estudos de Caso

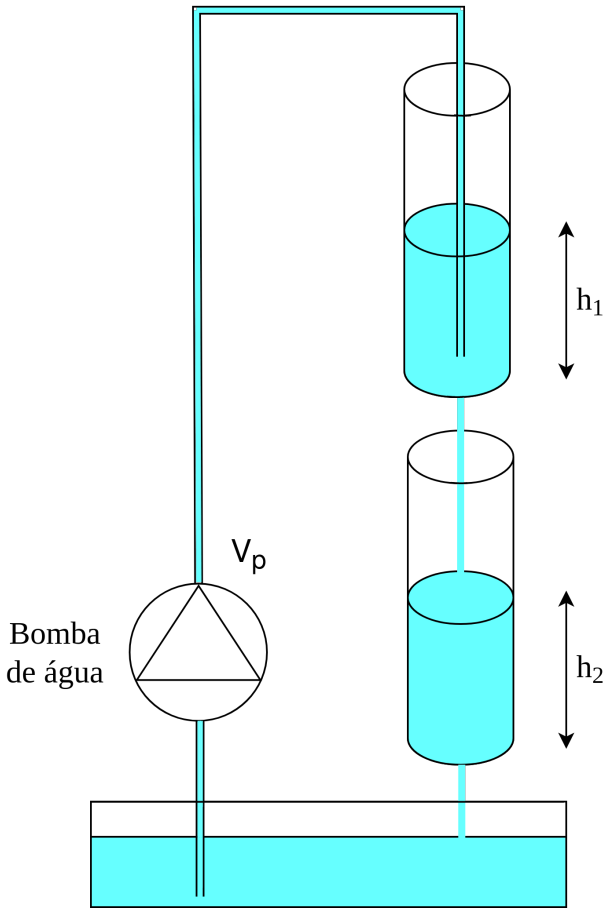
Nesse capítulo, apresentar-se-á brevemente cada sistema usado nos estudos de casos, a estratégia de otimização de controladores *fuzzy* utilizando a abordagem proposta, incluindo a escolha das faixas de frequência, seguido pelos resultados obtidos para cada estudo de caso adotado.

4.1 Sistema de Tanques Acoplados

O controle de nível de tanques é problema fundamental na indústria. Os processos industriais necessitam, frequentemente, que líquidos sejam bombeados e estocados em tanques, e oportunamente rebombeados para o outro tanque. Muitas vezes estes líquidos sofrerão processos químicos ou de mistura, de modo que será sempre necessário o controle do nível e do fluxo entre tanques. Por tanto, o controle de níveis de tanques é requisito básico na indústria química, setor essencial na economia, englobando a petroquímica, tratamento de água, indústria de produção do papel, dentre outros.

A planta sob análise é um sistema de tanques acoplados da Quanser (Figura 4.1), que possui dois tanques, um reservatório, e uma bomba hidráulica acionada por um motor DC, onde os dados são coletados por uma placa de aquisição de dados e enviados para um computador. Os sensores de níveis de cada tanque enviam sinais elétricos à placa de aquisição de dados, esta fornece os dados ao controlador e recebe do mesmo o sinal de controle para o acionamento da bomba. A bomba hidráulica fornece ao tanque 1, tanque mais alto, uma vazão diretamente proporcional a tensão aplicada ao seu motor. O líquido contido no tanque 1 flui por gravidade através de um orifício para o tanque 2. Do mesmo modo o líquido contido no tanque 2 flui para o reservatório, sendo possível o ajuste da vazão dos orifícios pela troca de peças. O controle do nível do tanque inferior será o abordado.

Figura 4.1: Sistema de tanques acoplados



Fonte: Autoria própria

Utilizando o balanço de massas de liquido para o sistema e aplicando a discretização utilizando Euler, com T sendo o período de amostragem, obtém-se o modelo discretizado da planta, dado pelas Equações 4.1 e 4.2.

$$h_1(k) = h_1(k-1) + T\left(\frac{K_m V_p}{A_2} - \frac{a_1\sqrt{2gV_p h_1(k-1)}}{A_1}\right) \tag{4.1}$$

$$h_2(k) = h_2(k-1) + T\left(\frac{a_1\sqrt{2gh_1(k)}}{A_2} - \frac{a_2\sqrt{2gh_2(k-1)}}{A_1}\right) \tag{4.2}$$

Sendo h_1 e h_2 níveis dos tanques 1 e 2 respectivamente, A_1 e A_2 as áreas das bases dos tanques 1 e 2 respectivamente, a_1 e a_2 são as áreas dos orifícios de saída dos tanques 1 e 2 respectivamente, K_m é a constante da bomba, enquanto V_p é a tensão aplicada à bomba e g a aceleração da gravidade.

Para a implementação foram adotados os parâmetros baseados em sistemas de tanques

da *Quanser*, conforme a tabela 4.1.

Tabela 4.1: Constantes usadas na modelagem do sistema de tanques acoplados

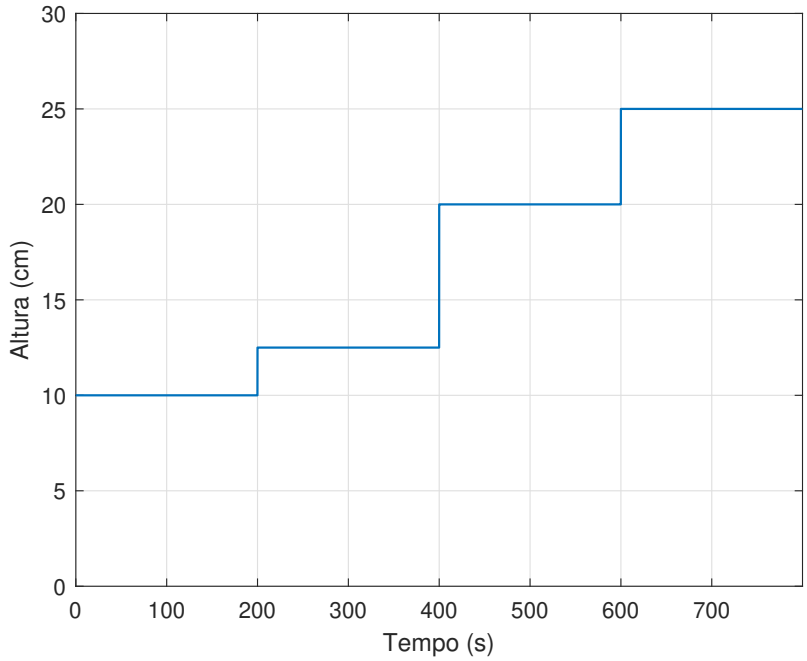
Símbolo	Descrição	Valor	Unidade
K_m	Constante de fluxo da bomba	3,3	$\frac{cm^2}{s \cdot V}$
a_1	Área do orifício de saída do tanque 1	0,178	cm
a_2	Área do orifício de saída do tanque 2	0,178	cm
A_1	Área da seção transversal do tanque 1	15,25	cm^2
A_2	Área da seção transversal do tanque 2	15,25	cm^2
g	Aceleração da gravidade	981	cm/s^2
T	Período de amostragem	0,1	s

Fonte: Autoria própria

4.1.1 Otimização do controlador

O controle do sistema de tanques acoplados, trata-se de um problema servo, tendo como entrada uma serie de degraus, conforme a Figura 4.2.

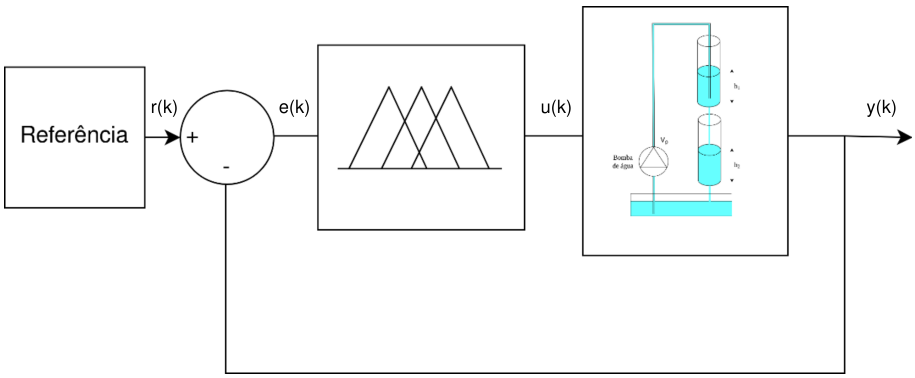
Figura 4.2: Referência para o controle de velocidade do motor DC



Fonte: Autoria própria

O esquema do sistema de controle é apresentado na Figura 4.3.

Figura 4.3: Sistema de controle para os tanques acoplados



Fonte: Autoria própria

O controlador a ser otimizado é um controlador *fuzzy*-PI Sugeno com três funções de pertinência de forma triangular para cada entrada. Para cada combinação possível entre as funções de pertinência das entradas, há uma regra e sua correspondente saída, totalizando nove regras e saídas. Os valores iniciais dos controladores foram sorteados em torno de valores conhecidos para o problema, a mesma população inicial foi utilizada para todos os algoritmos de otimização de modo que a aleatoriedade na definição da população inicial não influenciará na comparação das técnicas.

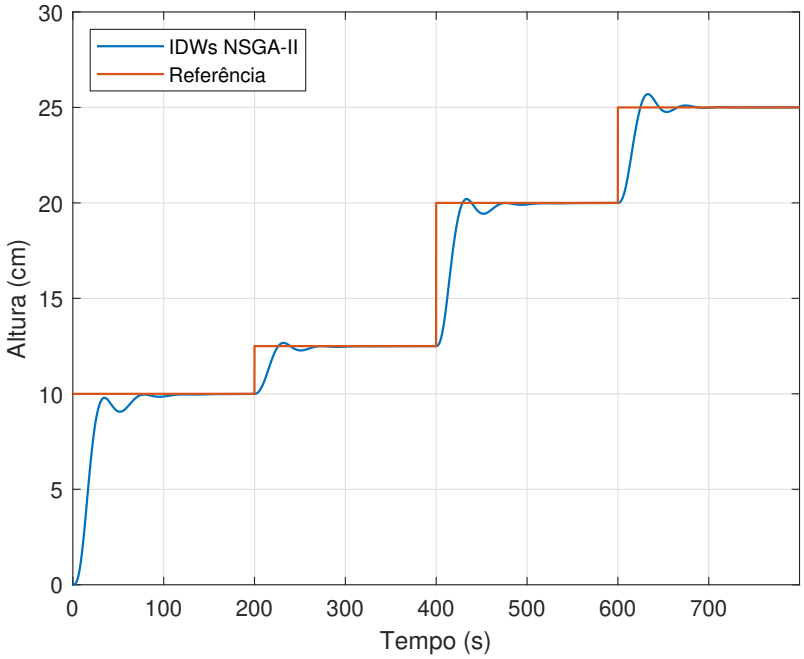
Na etapa de projeto, sabendo que a frequência de amostragem utilizada para a planta da Quanser é de $10Hz$, visando obter uma resposta rápida e com transitórios suaves, se particionou o erro em duas faixas de frequência. Como a frequência de mudança da referência, que é composta pela soma de degraus é de $0,002Hz$, adotou-se a faixa de frequência que descreve o comportamento no regime permanente de $0Hz$ e $0,005Hz$ e as frequências maiores, foram adotadas como aquelas que descrevem o comportamento do transitório.

Definido o problema, no domínio *wavelet*, foram adotadas essa duas faixas de valores de frequência para determinar dois $IDWs$ a serem otimizados usando o NSGA-II e NSGA-III. Para ambos os algoritmos de otimização, adotou-se uma população de 25 indivíduos executados por 20 gerações, o que foi determinado como critério de parada, valores esses obtidos de forma heurística. A porcentagem de cruzamento de 80%, enquanto a taxa de mutação é de 10%, também obtidos de forma heurística. Foram adotados, no NSGA-III, quatro pontos de referência definidos pelo algoritmo. Foi realizada, ainda, a sintonia do controlador pelo GA mono-objetivo, como o mesmo número de indivíduos e gerações, utilizando como função custo os índices IEA e IEAT apresentados na Seção 2.5, visando a comparação com a metodologia proposta.

4.1.2 Resultados

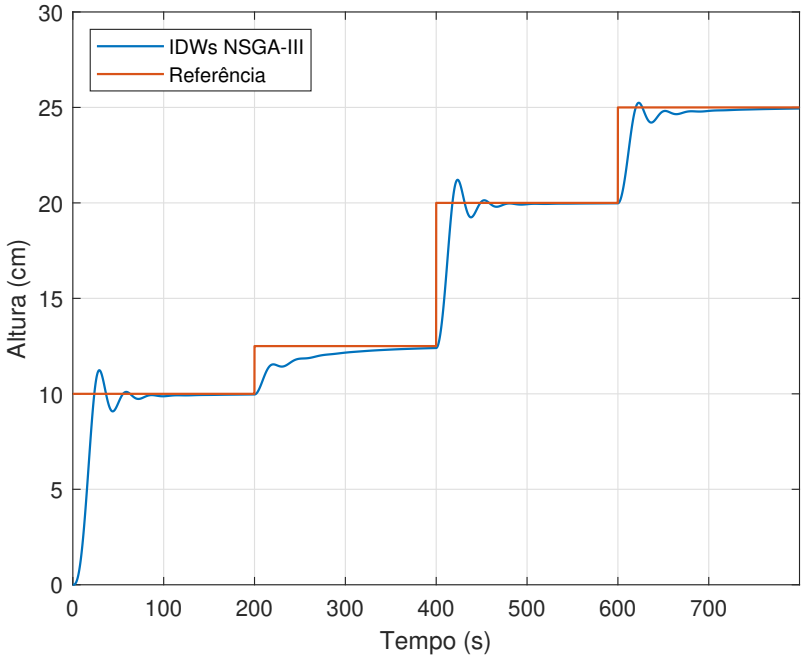
Utilizando-se da metodologia descrita, com as meta-heurísticas multiobjetivo NSGA-II e NSGA-III otimizando os $IDWs$ definidos, alcançou-se o resultado exposto na Figura 4.4 e na Figura 4.5.

Figura 4.4: Sinal de resposta referente aos *IDWs* otimizados com NSGA-II para o sistema de tanque



Fonte: Autoria própria

Figura 4.5: Sinal de resposta referente aos *IDWs* otimizados com NSGA-III para o sistema de tanque

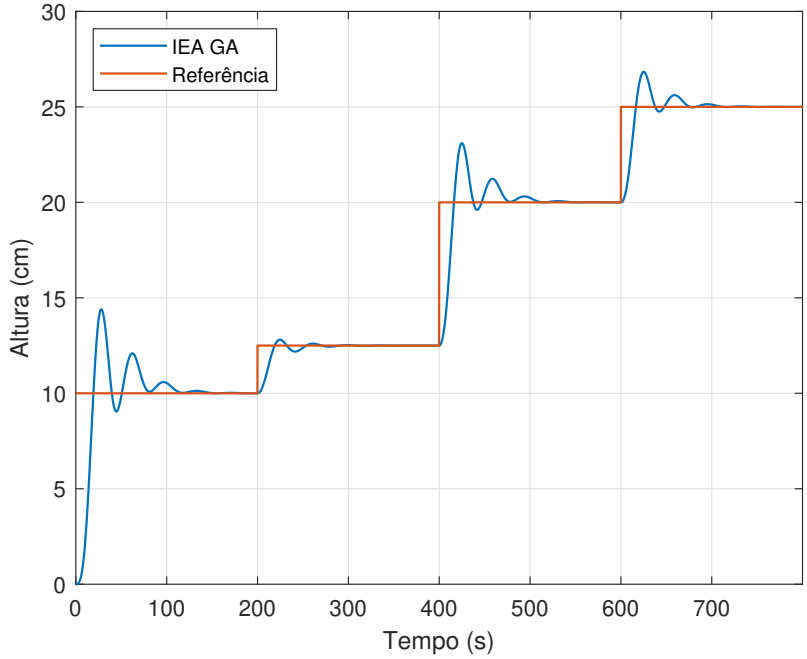


Fonte: Autoria própria

A resposta alcançada pelo controlador otimizado pelo NSGA-II segue bem a referência, e apresenta um transitório curto e sem grandes oscilações. Já a resposta apresentada pelo controlador otimizado pelo NSGA-III também segue bem a referência e também possui um transitório curto, porém com oscilações de maior amplitude em relação ao do NSGA-II.

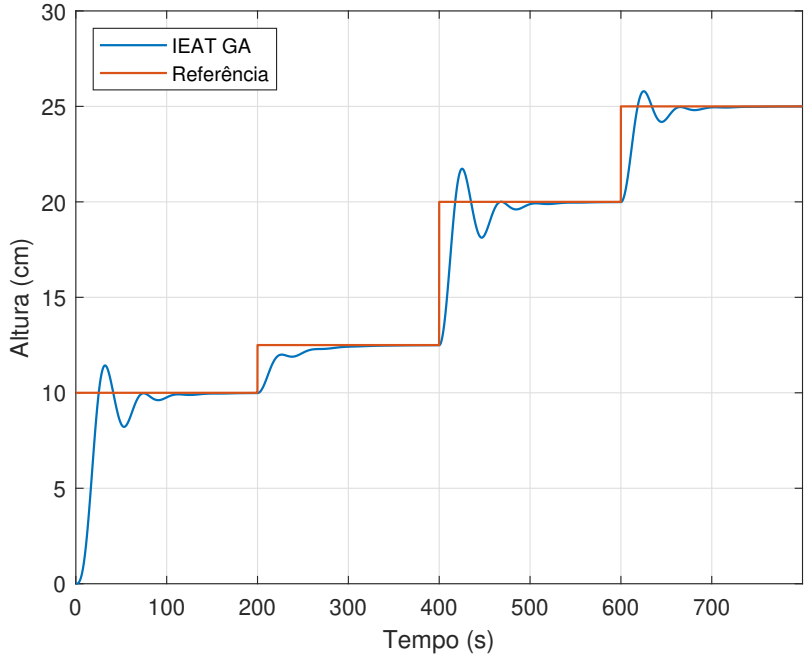
Para comparação com a técnica proposta se realizou a otimização de controle utilizando o GA mono-objetivo com o IEA e IEAT como função custo. Obteve-se, então, por simulação no Matlab a resposta exibida na Figura 4.6 para o IEA como função custo e o resultado apresentado na Figura 4.7 utilizando o IEAT como função custo. A comparação entre os resultados obtidos pelas técnicas está representado na Figura 4.8.

Figura 4.6: Sinal de resposta referente ao IEA para o sistema de tanque



Fonte: Autoria própria

Figura 4.7: Sinal de resposta referente ao IEAT para o sistema de tanque



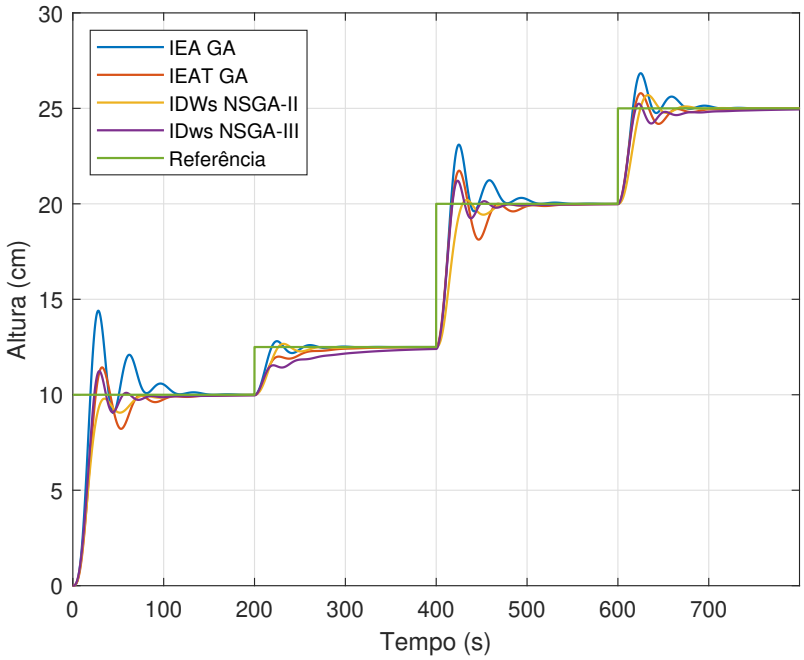
Fonte: Autoria própria

Ambas as respostas seguem bem a referência. O controlador encontrado usando GA

com IEA como função custo apresentou um transitório mais longo, com um maior número de oscilações com amplitudes significativas, o que se traduz em altos valores de *overshoot* e tempo de acomodação. O controlador encontrado usando GA com IEAT como função custo, apresentou um resposta com transitório curto e menores amplitudes de oscilação.

Calculando os valores de IEA para a resposta com cada um dos controladores otimizados tem-se: 13402cm.s para o GA utilizando IEA, 13074cm.s para o GA utilizando IEAT, 2903cm.s para os *IDWs* otimizados pelo NSGA-II e 13057cm.s para os *IDWs* otimizados pelo NSGA-III. Observando um melhor desempenho das técnicas propostas frente às mono-objetivo e um bom desempenho usando o IEAT, que se aproxima da otimização dos *IDWs* pelo NSGA-III e há um desempenho superior otimizando os índices propostos pelo NSGA-II.

Figura 4.8: Sinais de resposta referente aos quatro controladores para o sistema de tanque



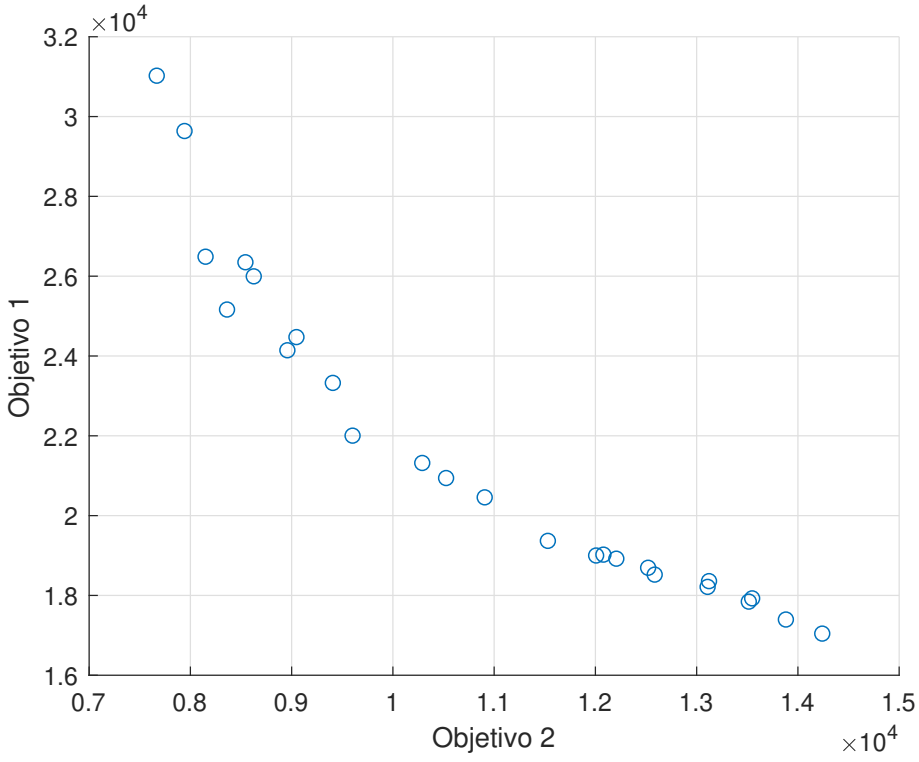
Fonte: Autoria própria

Comparando graficamente as respostas encontradas para os quatro controladores, observa-se que aquele sintonizado usando GA com IEAT como função custo, se assemelha bastante ao comportamento da resposta obtida otimizando os *IDWs* com NSGA-III, possuindo a técnica mono-objetivo, transitório um pouco menos comportado, podendo-se observar um tempo de acomodação um pouco maior e maiores amplitudes nas oscilações antes de se alcançar o regime permanente. Esses dois controladores também demoram mais para alcançar o regime permanente no segundo degrau da referência. Assim, a resposta encontrada usando o NSGA-II foi a melhor e a obtida usando GA com IEA como função custo, a pior, tanto levando em consideração o *overshoot*, o tempo de acomodação, a quantidade e a amplitude das oscilações no transitório e o tempo de acomodação. Desse modo, a resposta encontrada pelo controlador encontrado pelo NSGA-II foi tanto mais

rápida quanto mais suave.

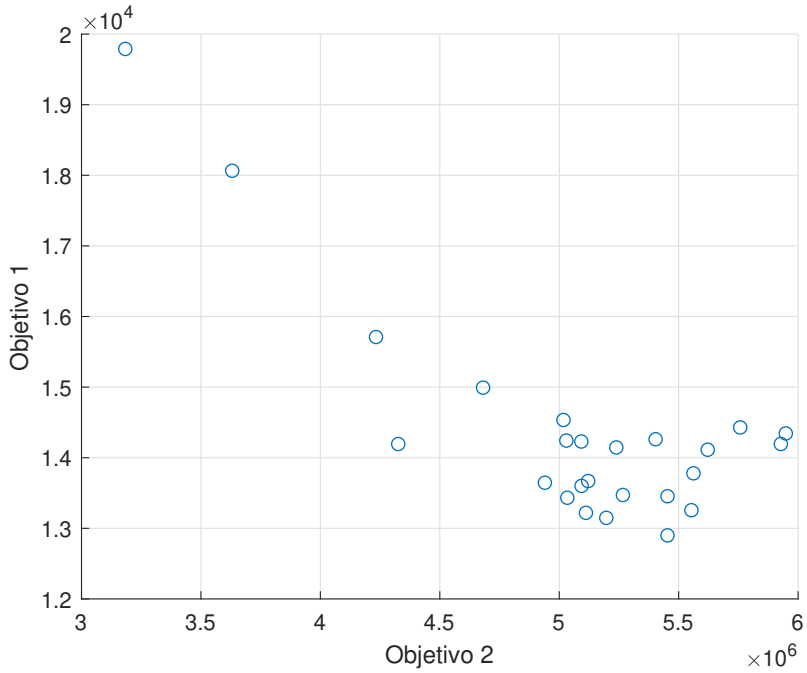
Para avaliar a otimização encontrada pelo NSGA-II e NSGA-III é importante apresentar a distribuição da população de soluções devolvida por cada um desses algoritmos, que idealmente aproximam a curva de Pareto para o problema, as distribuições para o NSGA-II e NSGA-III estão apresentadas respectivamente nas Figuras 4.9 e 4.9.

Figura 4.9: Distribuição da população do NSGA-II quanto aos objetivos



Fonte: Autoria própria

Figura 4.10: Distribuição da população do NSGA-III quanto aos objetivos



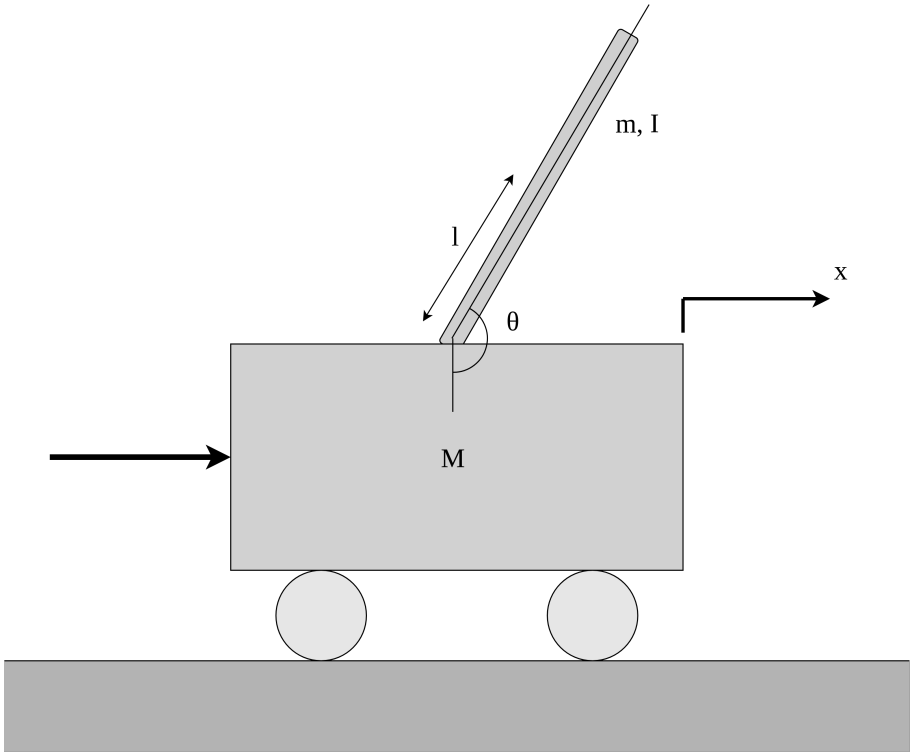
Fonte: Autoria própria

Pode-se observar que o NSGA-II apresentou uma população com soluções melhor distribuídas, quanto a cada um dos objetivos, apresentando um maior número de soluções não dominadas. Por outro lado, o NSGA-III conseguiu melhores resultados quanto ao primeiro objetivo, porém, apresentou soluções bem menos diversas, o que pode indicar que o algoritmo ficou preso em mínimos locais, observando-se que uma melhor escolha dos parâmetros para esse algoritmo poderia ser encontrada, como a taxa de mutação ou a escolha dos pontos iniciais para o nichamento mais adequados.

4.2 Pêndulo Invertido

O sistema sob análise nesse estudo de caso é um pêndulo invertido sob um carrinho motorizado, conforme TILBURY *et al.* (1998). O objetivo do sistema de controle é equilibrar o pêndulo invertido aplicando uma força ao carrinho ao qual o pêndulo está preso. Um exemplo do mundo real relacionado diretamente a esse sistema de pêndulo invertido é o controle de atitude de um foguete auxiliar na decolagem. No exemplo sob análise, o pêndulo é obrigado a se mover no plano vertical mostrado na figura abaixo. Para este sistema, a entrada de controle é a força F que move o carrinho horizontalmente e as saídas são a posição angular $Theta$ do pêndulo e a posição horizontal do carrinho x.

Figura 4.11: Pêndulo invertido



Fonte: Autoria própria

Utilizou-se os seguintes valores para os parâmetros do sistema:

Tabela 4.2: Valores de parâmetros usados na modelagem do sistema de pêndulo invertido

Símbolo	**Descrição**	**Valor**	**Unidade**
M	Massa do carrinho	0,5	kg
m	Massa do pêndulo	0,2	kg
b	Coeficiente de atrito para o carrinho	0,1	N/m/s
l	Comprimento até o centro de massa do pêndulo	0,3	m
I	Momento de inércia de massa do pêndulo	0,006	kg.m^2

Fonte: Autoria própria

A Figura 4.12 mostra o diagrama de corpo livre dos dois elementos do sistema de pêndulo invertido.

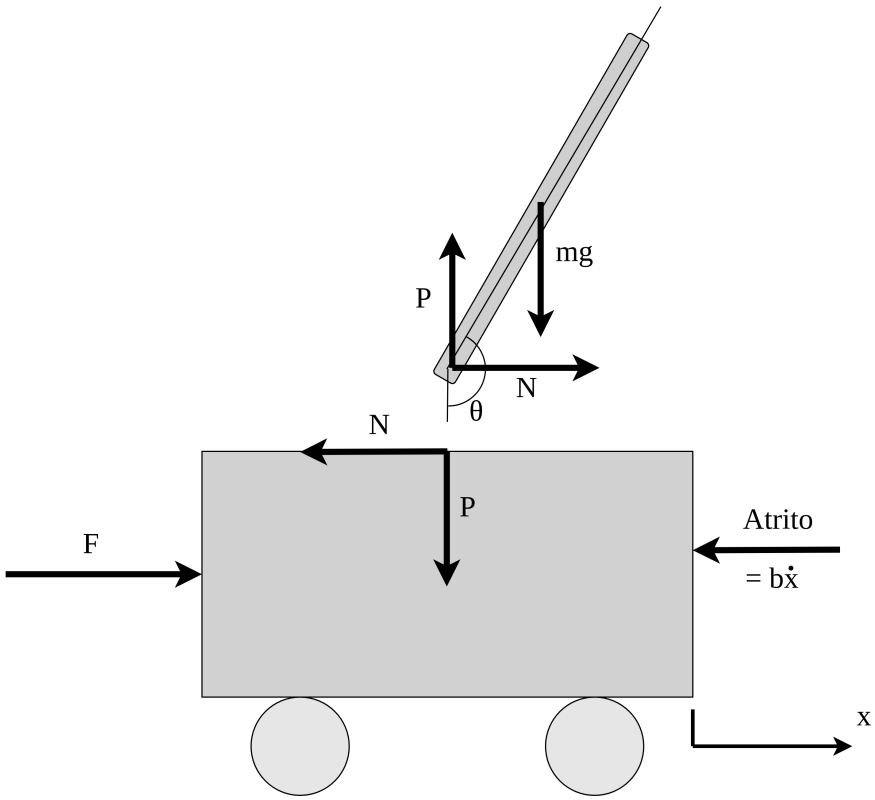


Figura 4.12: Diagrama de corpo livre do sistema de pêndulo invertido

Fonte: Autoria própria

Usando a soma das forças do carrinho na direção horizontal no diagrama de corpos livres, obtém-se a seguinte equação de movimento:

$$M\ddot{x} + b\dot{x} + N = F \tag{4.3}$$

Somando as forças no diagrama de corpo livre do pêndulo na direção horizontal, obtém-se a seguinte expressão para a força horizontal aplicada pelo carrinho no pêndulo, equivalente à força de reação N:

$$N = m\ddot{x} + ml\ddot{\theta}cos\theta - ml\dot{\theta}^2 sen\theta \tag{4.4}$$

Substituindo-se a equação 4.4 na equação 4.3, obtém-se uma das equações de movimento para o sistema:

$$(M+N)\ddot{x} + b\dot{x} + ml\ddot{\theta}cos\theta - ml\dot{\theta}^2 sen\theta = F \tag{4.5}$$

Para obter a segunda equação de movimento, pode-se somar as forças perpendiculares ao pêndulo, obtém-se, então:

$$P\,sen\theta + N\,cos\theta - mg\,sen\theta = ml\ddot{\theta} + m\ddot{x}cos\theta \tag{4.6}$$

Para eliminar os termos P e N, que representam a força entre o carrinho e o pêndulo,

na equação acima, soma-se os momentos em relação ao centroide do pêndulo obtendo a seguinte equação:

$$-Pl\,sen\theta + Nl\,cos\theta = I\ddot{\theta} \tag{4.7}$$

Combinando as equações 4.6 e 4.7, obtém-se a segunda equação:

$$(I + ml^2)\ddot{\Theta} + mgl\,sen\theta = -ml\ddot{x}\,cos\theta \tag{4.8}$$

Linearizando as equações tomando a posição em que o pêndulo está equilibrado verticalmente para cima, logo $\theta = \phi$, sendo ϕ o desvio da posição do pêndulo em relação ao equilíbrio, ou seja, $\theta = \pi + \phi$ e presumindo um pequeno desvio (ϕ) do equilíbrio, caso em que podemos usar as seguintes aproximações de pequenos ângulos das funções não lineares nas equações de sistema:

$$\begin{aligned} cos\theta &= cos(\pi + \phi) \approx -1 \\ sen\theta &= sen(\pi + \phi) \approx -\phi \\ \dot{\theta}^2 &= \dot{\phi}^2 \approx 0 \end{aligned} \tag{4.9}$$

Substituindo as aproximações acima em nossas equações governantes não-lineares, tendo como entrada u a força aplicada F, ou seja, $u = F$, pode-se obter:

$$(I + ml^2)\ddot{\phi} - mgl\phi = ml\ddot{x} \tag{4.10}$$

$$(M + m)\ddot{x} + b\dot{x} - ml\ddot{\phi} = u \tag{4.11}$$

Realizando a transformada de Laplace das equações do sistema assumindo condições iniciais nulas, obtém-se:

$$(I + ml^2)\Phi(s)s^2 - mgl\Phi(s) = mlX(s)s^2 \tag{4.12}$$

$$(M + m)X(s)s^2 + bX(s)s - ml\Phi(s)s^2 = U(s) \tag{4.13}$$

Para encontrar as funções de transferência para a saída $\phi(s)$ e uma entrada de $U(s)$, elimina-se $X(s)$ das equações acima. Colocando a equação 4.13 em função de $X(s)$, tem-se:

$$X(s) = \left[\frac{I + ml^2}{ml} - \frac{g}{s^2}\right]\Phi(s) \tag{4.14}$$

Substituindo a equação 4.14 na equação 4.13 chega-se a:

$$(M + m)\left[\frac{I + ml^2}{ml} - \frac{g}{s^2}\right]\Phi(s)s^2 + b\left[\frac{I + ml^2}{ml} - \frac{g}{s^2}\right]\Phi(s)s - ml\Phi(s)s^2 = U(s) \tag{4.15}$$

Reorganizando os termos chega-se a função de transferência:

$$\frac{\Phi(s)}{U(s)} = \frac{\frac{ml}{q}s^2}{s^4 + \frac{b(I+ml^2)}{q}s^3 - \frac{(M+m)mlg}{q}s^2 - \frac{bmgl}{q}s} \tag{4.16}$$

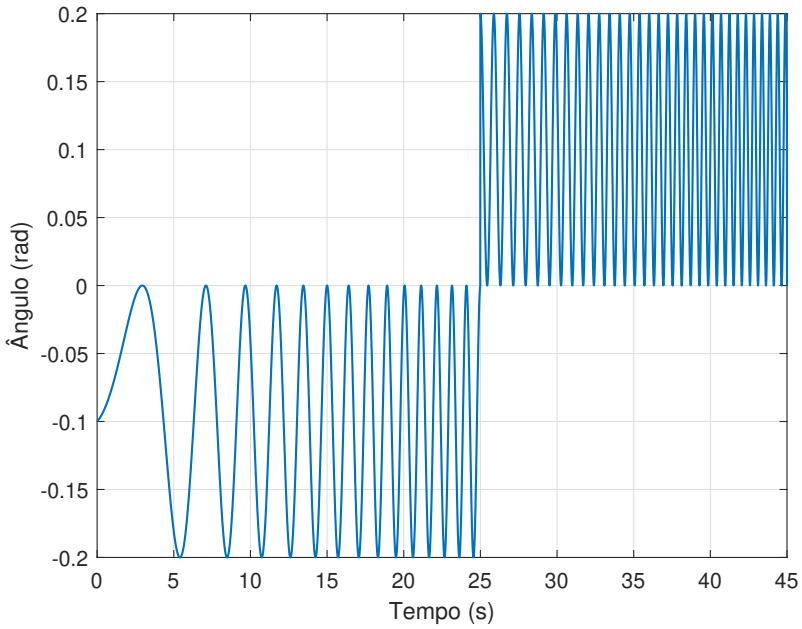
em que,

$$q = \left[(M+m)\left(I+ml^2\right) - (ml)^2\right] \tag{4.17}$$

4.2.1 Otimização do controlador

O controle do sistema de pêndulo invertido, trata-se de um problema regulador, tendo como perturbação o sinal apresentado na Figura 4.13 com referência mantida em zero.

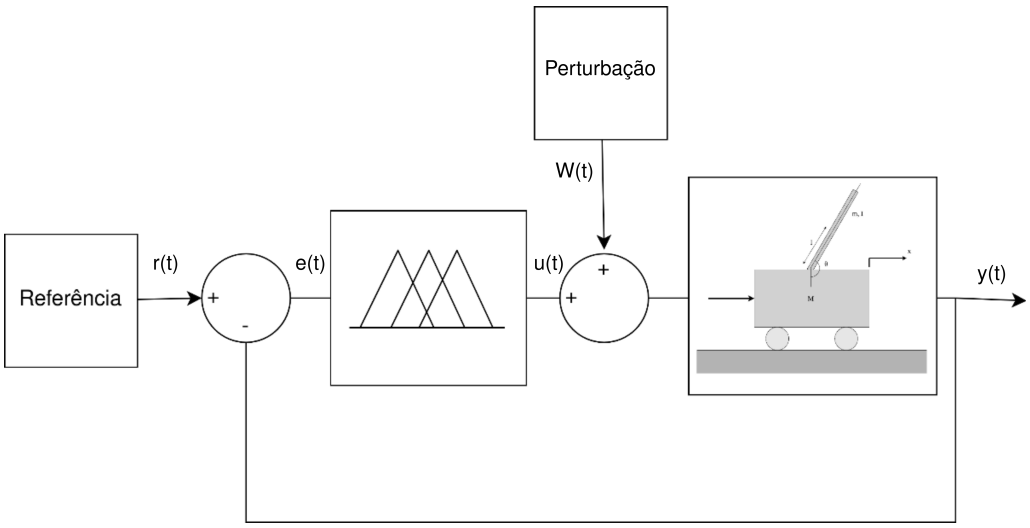
Figura 4.13: Pertubação no sistema de pendulo invertido



Fonte: Autoria própria

O esquema do sistema de controle é apresentado na Figura 4.14.

Figura 4.14: Sistema de controle para o pendulo invertido



Fonte: Autoria própria

O controlador a ser otimizado é um controlador *fuzzy*-PID Sugeno com três funções de pertinência de forma triangular para cada entrada. Para cada combinação possível entre as funções de pertinência das entradas, há uma regra e sua correspondente saída, totalizando vinte e sete regras e saídas. Os valores iniciais dos controladores foram sorteados em torno de valores conhecidos para o problema, a mesma população inicial foi utilizada para todos os algoritmos de otimização de modo que a aleatoriedade na definição da população inicial não influenciará na comparação das técnicas.

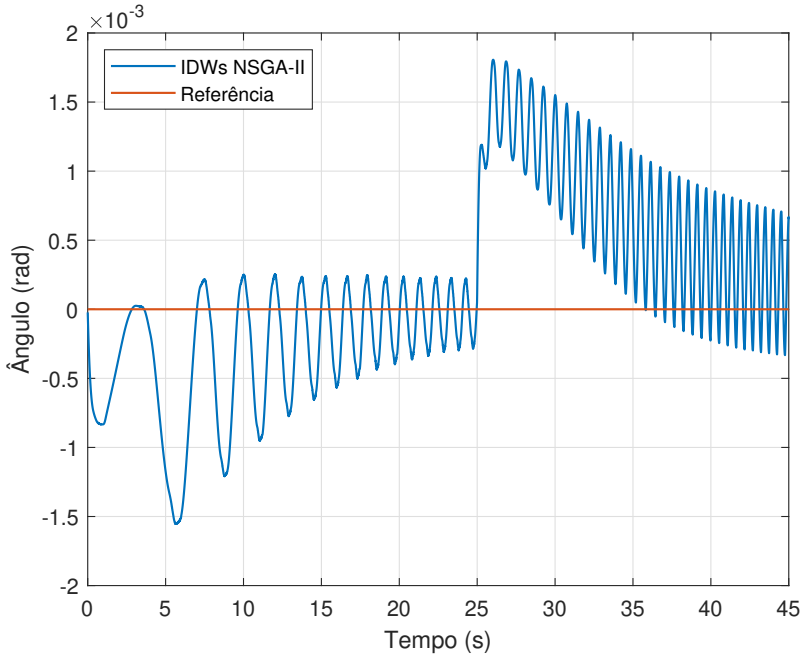
A frequência de amostragem do sistema é de $10^3 Hz$, visando a rejeição de uma perturbação com múltiplas frequências como a apresentada na Figura 4.13. Para o projeto dos $IDWs$ é possível particionar o problema em várias faixas de frequências devido às características da referência, em que as frequências aumentam com passar do tempo, porém, de forma experimental se fixou a definição em três faixas de frequência. A primeira faixa de $0Hz$ a $0,8Hz$, a segunda de $0,9Hz$ a $1,6Hz$ e a terceira de $1.7Hz$ a $500Hz$.

Definido os $IDWs$ a serem otimizados usando os algoritmos de otimização multiobjetivo, definiu-se de forma heurística, para ambas as técnicas de otimização, uma população de 25 indivíduos, e 20 gerações como critério de parada. A porcentagem de cruzamento de 80% e a taxa de mutação é de 5% foram definidas de forma experimental. Foram adotados, no NSGA-III, quatro pontos de referências definidos pelo algoritmo. Para viabilizar uma comparação, foi realizada a sintonia do controlador pelo GA, utilizando como função custo os índices IEA e IEAT apresentados na Seção 2.5.

4.2.2 Resultados

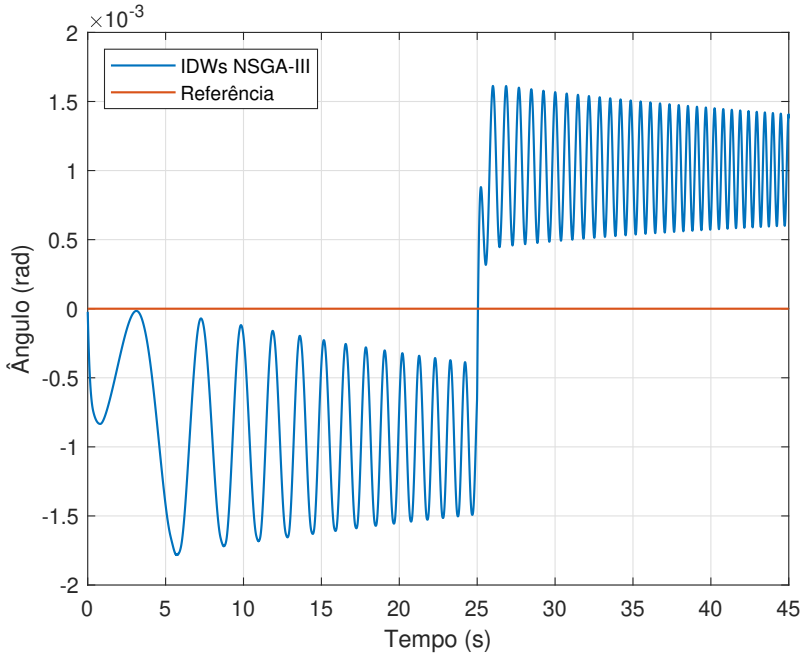
Fazendo uso da metodologia adotada, com o emprego do erro decomposto por transformada *wavelet* como função custo para a otimização usando NSGA-II e NSGA-III, alcançou-se os resultados expostos, respectivamente, na Figura 4.4 e na Figura 4.16.

Figura 4.15: Sinal de resposta referente aos *IDWs* otimizados com NSGA-II para o pendulo invertido



Fonte: Autoria própria

Figura 4.16: Sinal de resposta referente aos *IDWs* otimizados com NSGA-III para o pendulo invertido

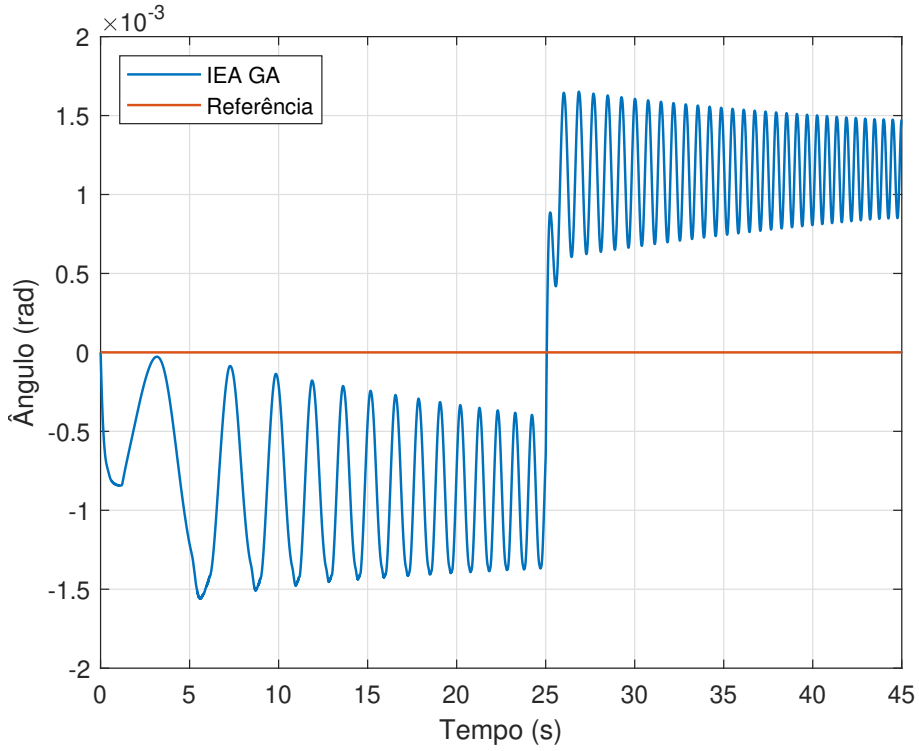


Fonte: Autoria própria

Ambas as respostas conseguem rejeitar a perturbação, reduzindo sua ação sobre a resposta, de modo que o sistema não se distancie significativamente da referência e mantendo a estabilidade. O controlador encontrado usando o NSGA-II apresentou uma resposta ligeiramente melhor, pois além de reduzir a amplitude das oscilações, nas diversas frequências impostas pela perturbação, consegue reduzir rapidamente o *offset* causado pelos degraus presentes na pertubação, ao contrário do NSGA-III.

Com a finalidade de comparação com a técnica proposta, foi realizada a otimização de controle utilizando o GA mono-objetivo com o IEA e IEAT como função custo. Obteve-se, então, por simulação no Matlab a resposta exibida na Figura 4.17 para o IEA e Figura 4.18 para o IEAT. A comparação entre os resultados obtidos pelas técnicas está representado na Figura 4.19.

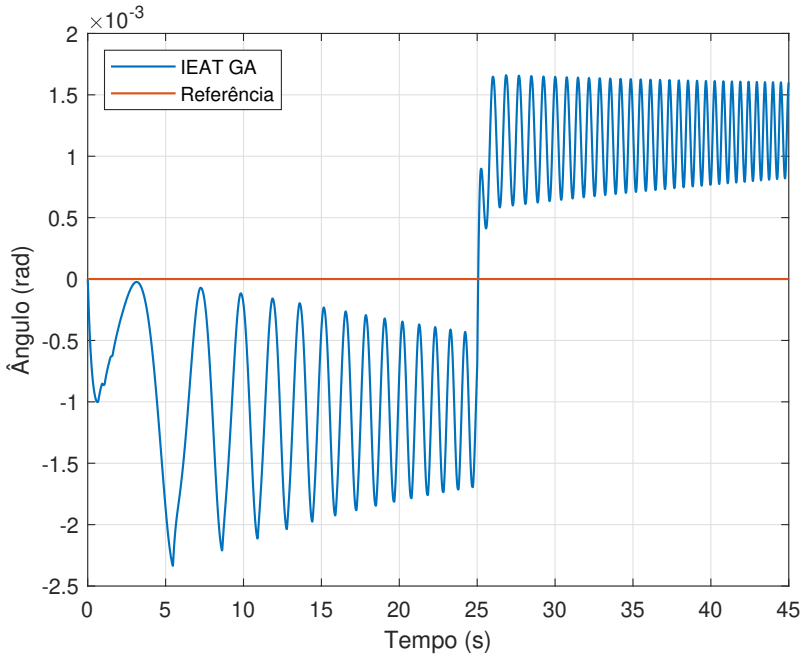
Figura 4.17: Sinal de resposta referente ao IEA para o pendulo invertido



Fonte: Autoria própria

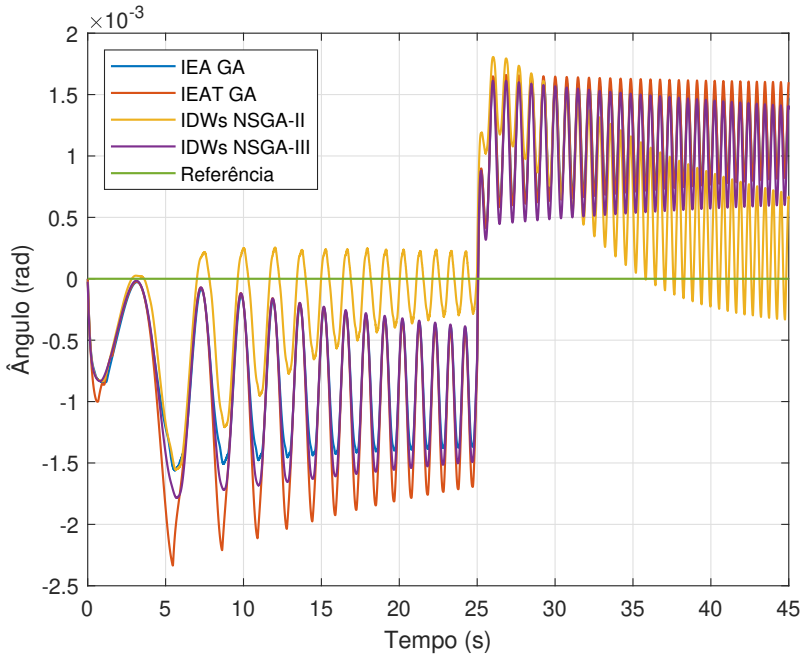
Calculando os valores de IEA para a resposta com cada um dos controladores otimizados tem-se: 0,0436rad.s para o GA utilizando IEA, 0,0476rad.s para o GA utilizando IEAT, 0.0248rad.s para os *IDWs* otimizados pelo NSGA-II e 0.0423rad.s para os *IDWs* otimizados pelo NSGA-III. Observa-se um melhor desempenho das técnicas propostas frente às mono-objetivo havendo um desempenho superior na otimização utilizando os índices propostos com NSGA-II.

Figura 4.18: Sinal de resposta referente ao IEAT para o pendulo invertido



Fonte: Autoria própria

Figura 4.19: Sinais de resposta dos quatro controladores para o pendulo invertido



Fonte: Autoria própria

Graficamente, a resposta obtida com o controlador encontrado usando GA com IEA

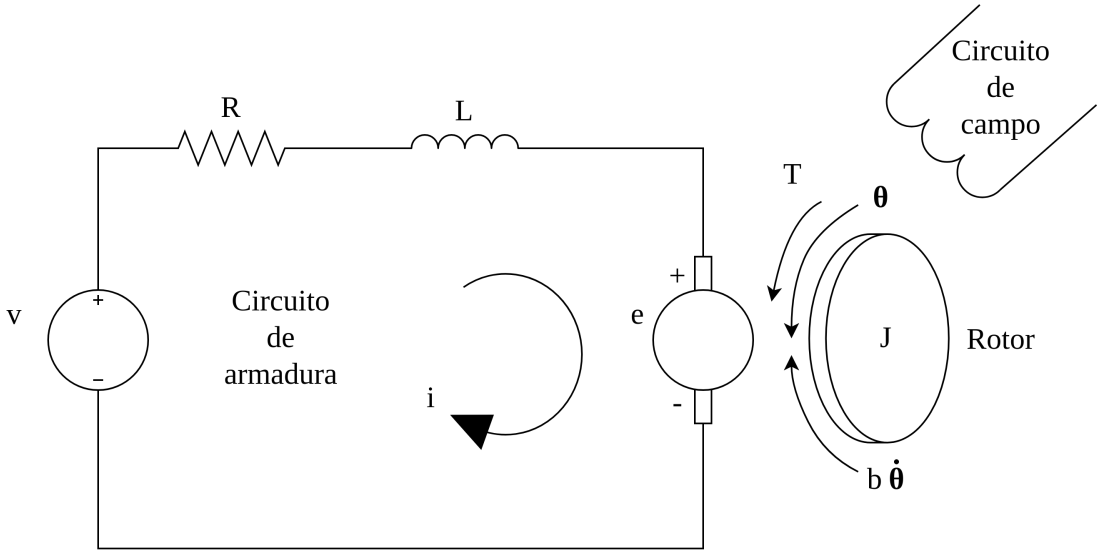
como função custo e com IEAT como função custo assemelham-se a encontrada pelo otimizado pelo NSGA-III, contudo a que usou como custo o IEA apresenta menores amplitudes de oscilação frente a encontrada pelo NSGA-III, que, por sua vez se sobressai, nesse quesito, sobre o encontrado usando como função custo o IEAT. Ademais, a resposta alcançada pelo controlador sintonizado pelo NSGA-II apresentou o melhor resultado rejeitando o degrau presente na perturbação.

Como foram utilizadas três funções objetivo para otimização, não será apresentada a distribuição da população de soluções no espaço das funções objetivo pela dificuldade de representação de pontos no espaço tridimensional na forma impressa.

4.3 Motor-DC: Controle da posição

O motor DC é um atuador comumente usado em sistemas de controle, tanto fornecendo um movimento rotativo, quanto acoplado a rodas, engrenagens, cabos ou outros equipamentos. O sistema mostrado e abordagem dada está de acordo com TILBURY *et al.* (1998). Para a modelagem desse sistema deve-se analisar tanto o circuito elétrico equivalente da armadura do motor quanto o diagrama de corpo livre do rotor, esse sistema é mostrado na figura 4.20.

Figura 4.20: Motor-DC



Fonte: Autoria própria

No caso em análise, vamos assumir os valores para os parâmetros físicos obtidos por experimento de um motor real no laboratório de controles de graduação da Carnegie Mellon, segundo TILBURY *et al.* (1998). Os valores estão expostos no seguinte quadro:

Tabela 4.3: Valores de parâmetros usados na modelagem do sistema de motor-DC

Símbolo	Descrição	Valor	Unidade
J	Momento de inércia do rotor	3,2284E-6	kg.m^2
b	Constante de atrito viscoso do motor	3,5077E-6	N.m.s
K_b	Constante de força eletromotriz	0,0274	V/rad/seg
K_t	Constante de torque do motor	0,0274	N.m/Amp
R	Resistência elétrica	4	Ohm
L	Indutância elétrica	2,75E-6	H

Fonte: Autoria própria

Em regra, o torque produzido por um motor DC é proporcional à corrente de armadura e à força do campo magnético. Assumindo campo magnético constante e, portanto, que o torque do motor é proporcional à corrente de armadura i por um fator constante K_t, ou seja, um motor controlado por armadura temos a equação 4.18.

$$T = K_t i \tag{4.18}$$

Como a força contra-eletromotriz , e, é proporcional à velocidade angular do eixo por um fator constante K_b, tem-se:

$$e = K_b \dot{\theta} \tag{4.19}$$

Como o torque do motor e as constantes de força contra-eletromotriz são iguais, isto é, $K_t = K_e$, pode-se adotar uma única constante K para representar o torque do motor e a constante da força contra-eletromotriz. Com esta consideração, a partir da análise da figura 4.20, pode-se obter as equações 4.20 e 4.21 com base na 2ª lei de Newton e na lei de tensão de Kirchhoff.

$$J\ddot{\theta} + b\dot{\theta} = K_i \tag{4.20}$$

$$L\frac{di}{dt} + R_i = V - K\dot{\theta} \tag{4.21}$$

Aplicando a transformada de Laplace às equações 4.20 e 4.21 , sendo s a variável de Laplace, obtemos as 4.22 e 4.23.

$$s\,(Js+b)\,\Theta(s) = KI(s) \tag{4.22}$$

$$(Ls+R)\,I(s) = V(s) - K\Theta(s) \tag{4.23}$$

Isolando $I(s)$ nas equações 4.22 e 4.23 e igualando os termos obtém-se equações 4.24.

$$I(s) = \frac{s\,(Js+b)\,\Theta(s)}{K} = \frac{V(s) - K\Theta(s)}{(Ls+R)} \tag{4.24}$$

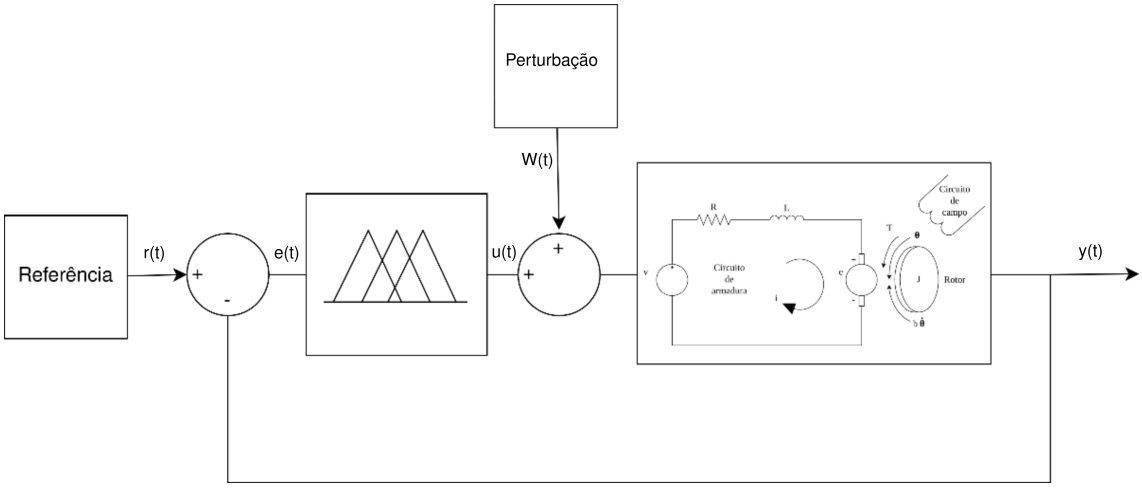
Então, considerando a velocidade de rotação $\dot{\Theta}(s)$ como saída e a tensão da armadura $V(s)$ como entrada, pode-se obter a função de transferência para o motor DC, expresso na equação 4.23.

$$\frac{\dot{\Theta}(s)}{V(s)} = \frac{K}{(Js+b)(Ls+R)+K^2} \quad \left[\frac{rad/sec}{V}\right] \tag{4.25}$$

4.3.1 Otimização do controlador

O controle do sistema de tanques acoplados, trata-se de um problema servo, porém com perturbação, tendo como entrada um degrau unitário e como perturbação uma senoide com amplitude de $0,5$ e frequência de $10 rad/sec$, ou seja, de $1,5915 Hz$. O esquema do sistema de controle é apresentado na Figura 4.21.

Figura 4.21: Sistema de controle da posição do motor DC



Fonte: Autoria própria

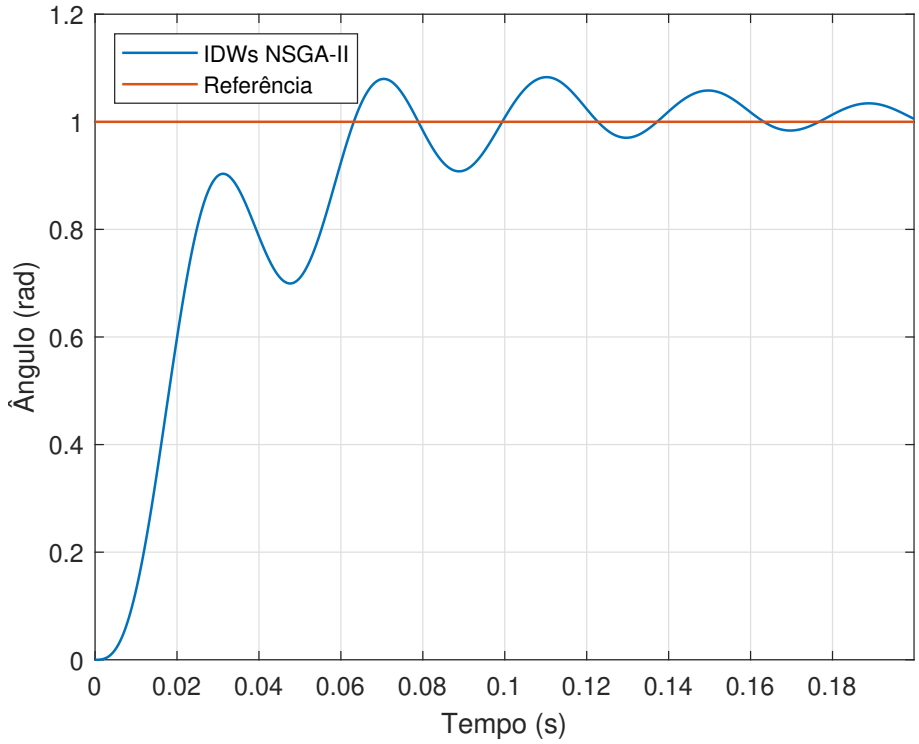
O controlador a ser otimizado é um controlador *fuzzy*-PI Sugeno com três funções de pertinência de forma triangular para cada entrada. Para cada combinação possível entre as função de pertinência das entradas, há uma regra e sua correspondente saída, totalizando nove regras e saídas. Os valores iniciais dos controladores foram sorteados em torno de valores conhecidos para o problema, a mesma população inicial foi utilizada para todos os algoritmos de a otimização de modo que a aleatoriedade na definição da população inicial não influenciará na comparação das técnicas.

Na etapa de projeto, para definir os $IDWs$, foi utilizada a abordagem experimental, sendo escolhidas duas faixas de frequência: de $0Hz$ a $18Hz$ e de $19Hz$ a $500Hz$. Definido o problema no domínio *wavelet* foi realizada a otimização usando o NSGA-II e NSGA-III. De forma heurística determinou-se uma população de 25 e 10 gerações como critério de parada, para os algoritmos de otimização. A porcentagem de cruzamento de 80% e taxa de mutação é de 5% foram adotadas para ambas as técnicas multiobjetivo e para o NSGA-III foram definidos seis pontos de referências calculados pelo algoritmo. Também foi realizada a sintonia do controlador pelo GA mono-objetivo utilizando IEA e IEAT para comparação.

4.3.2 Resultados

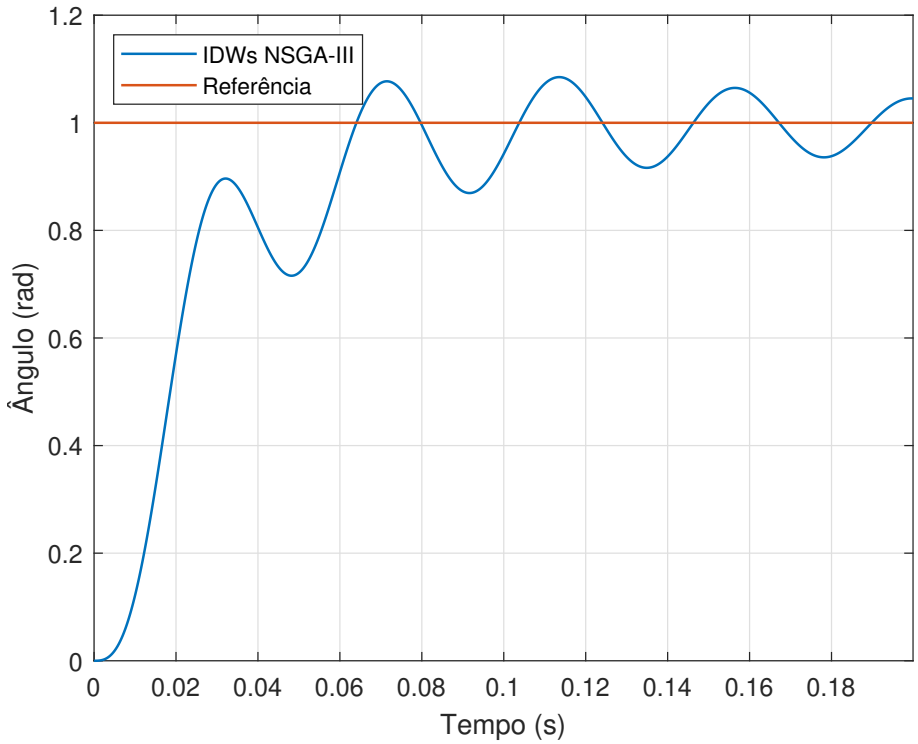
Para validação da metodologia descrita, foi realizada a simulação no Matlab utilizando as meta-heurísticas multi-objetivo adotadas para a otimização empregando o erro decomposto por transformada *wavelet* como função custo, obteve-se o resultado exposto na Figura 4.22 para o NSGA-II e na Figura 4.23 para o NSGA-III.

Figura 4.22: Sinal de resposta referente aos *IDWs* otimizados com NSGA-II para o controle de posição do motor DC



Fonte: Autoria própria

Figura 4.23: Sinal de resposta referente aos *IDWs* otimizados com NSGA-III para o controle de posição do motor DC

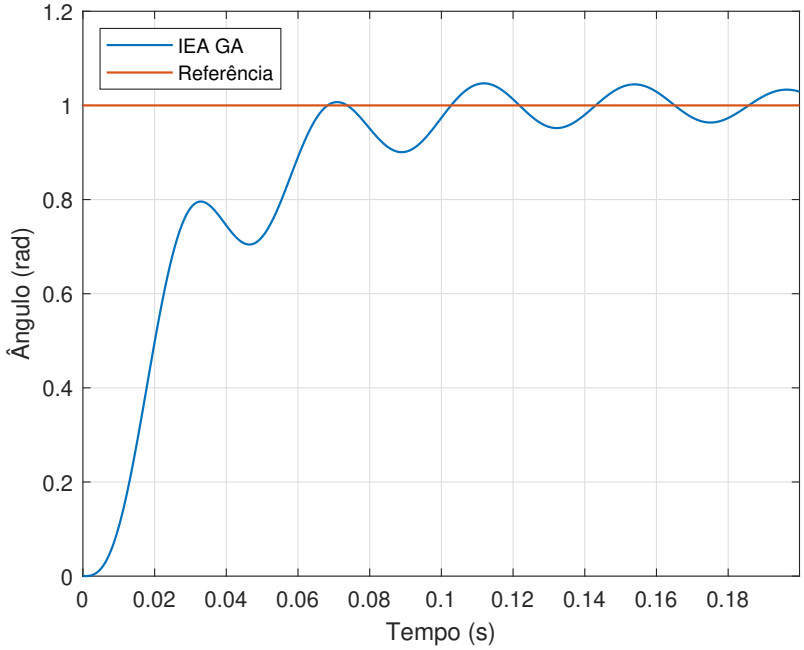


Fonte: Autoria própria

As respostas encontradas seguem a referência e reduzem as oscilações, que pelas frequências e amplitudes apresentadas aparentam ser causadas pela perturbação. O controlador encontrado pelo NSGA-II reduz as oscilações de modo mais eficaz, de modo que a resposta encontrada apresenta amplitudes e frequências progressivamente menores frente ao encontrado pelo NSGA-III.

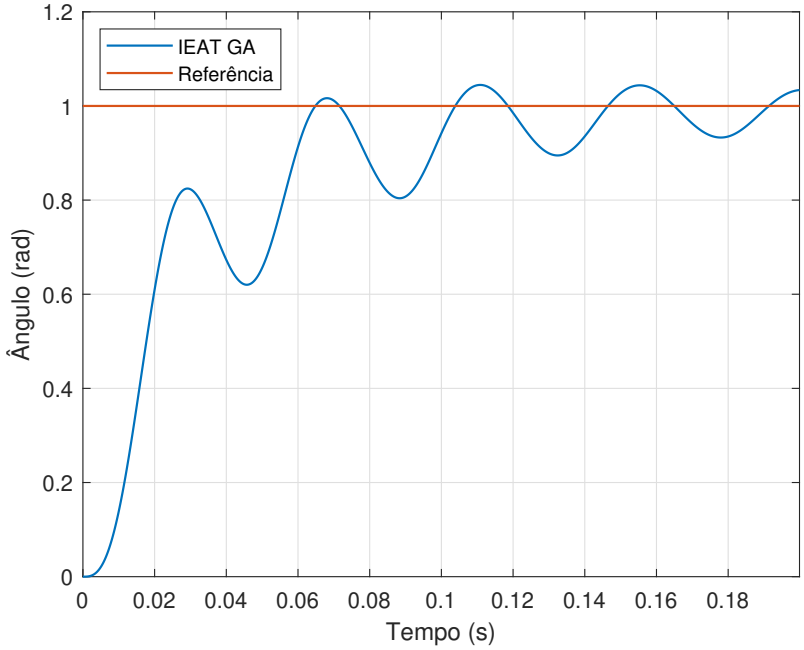
Para comparação, foram otimizados controladores usando o GA mono-objetivo com o IEA e IEAT como funções custo. Obteve-se por simulação no Matlab a resposta exibida na Figura 4.24 para o IEA e na Figura 4.25 para o IEAT. A comparação entre os resultados será apresentado na Figura 4.26.

Figura 4.24: Sinal de resposta referente ao IEA para o controle de posição do motor DC



Fonte: Autoria própria

Figura 4.25: Sinal de resposta referente ao IEAT para o controle de posição do motor DC

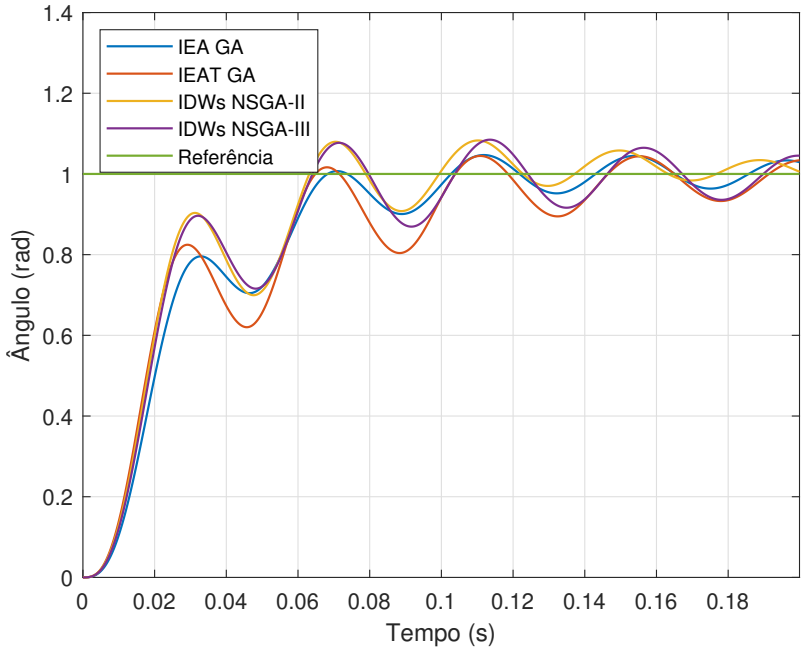


Fonte: Autoria própria

As respostas apresentada pelos controladores encontradas pela técnica mono-objetivo

seguem bem a referência, apresentando resposta suave e com razoável rejeição da perturbação. Porém, o controlador encontrado usando IEAT como função custo, apresentou um resposta com oscilações de maior amplitude, o que pode ser interpretado como uma rejeição menos eficaz a perturbação.

Figura 4.26: Sinais de resposta para os quatro controladores de posição para o motor DC

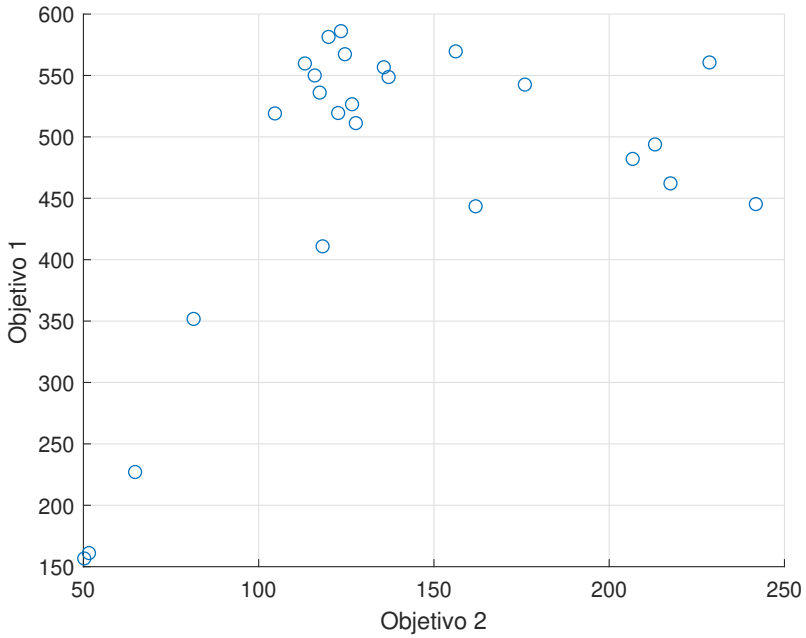


Fonte: Autoria própria

Calculando os valores de IEA para a resposta com cada um dos controladores otimizados tem-se: 0,1711rad para o GA utilizando IEA, 0,1677rad para o GA utilizando IEAT, 0,1773rad para os *IDWs* otimizados pelo NSGA-II e 0,1741rad para os *IDWs* otimizados pelo NSGA-III. Observa-se a obtenção de valores muito próximos, sendo que as técnicas mono-objetivo tiveram um melhor desempenho quanto ao IEA. Comparando graficamente as respostas encontradas para os quatro controladores, o sintonizado usando os índices propostos otimizados pelo NSGA-II, apresenta resposta semelhante ao encontrado pelo NSGA-III, apresentando ambos uma melhor rejeição a pertubação e respostas mais rápidas frente aos demais controladores, o que pode se traduzir na redução do erro absoluto. Assim, os controladores sintonizados usando o GA mono-objetivo apresentam um seguimento de referência mais lento, sendo que o IEAT apresentou pior desempenho para rejeitar as oscilações. Utilizando os mesmos critérios, há, ainda, vantagem do NSGA-II frente ao NSGA-III. Assim, observa-se que embora haja um pior índice numérico as respostas encontradas pela técnica proposta, graficamente foram superiores às técnicas mono-objetivo.

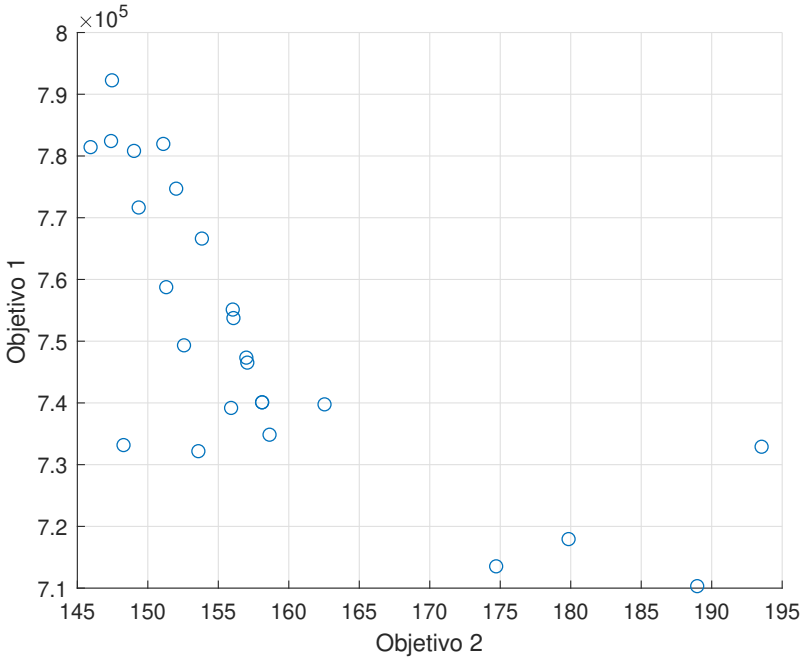
Para avaliar a otimização realizada pelos algoritmos multiobjetivo adotados será apresentada a distribuição da população de soluções encontradas por cada um desses algoritmos, as distribuições para o NSGA-II e NSGA-III estão apresentadas respectivamente nas Figuras 4.27 e 4.28.

Figura 4.27: Distribuição da população do NSGA-II quanto aos objetivos



Fonte: Autoria própria

Figura 4.28: Distribuição da população do NSGA-III quanto aos objetivos



Fonte: Autoria própria

Pode-se observar que o NSGA-III apresentou uma população com soluções melhor

distribuídas, quanto a cada um dos objetivos. Por outro lado, o NSGA-II conseguiu melhores resultados quanto a ambos os objetivos, embora corra o risco de ficar preso em mínimos locais se o algoritmo realizasse mais gerações, indicando que poderia ter sido escolhida uma maior taxa de mutação para o NSGA-II.

4.4 Motor-DC: Controle da velocidade

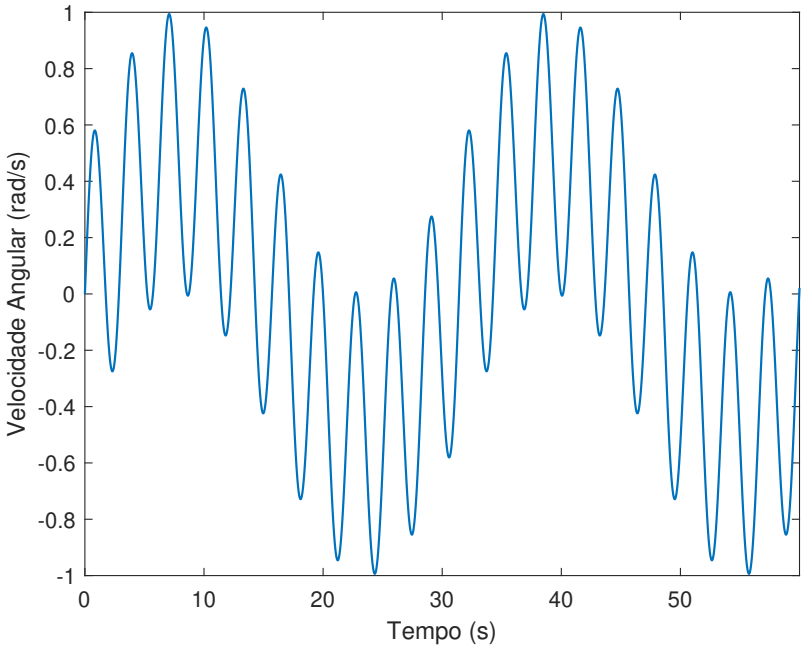
Neste caso, se abordará o motor-DC apresentado por (TILBURY *et al.*, 1998). Nesta analise se tomará como saída a posição. Como o sistema é idêntico ao da seção 4.3, pode-se obter a posição integrando a velocidade angular, logo, basta dividir a função de transferência apresentada na equação 4.25 por s, obtendo a função de transferência para o caso em análise, que tem por saída a posição $\Theta(s)$ e por entrada a tenção de armadura $V(s)$.

$$\frac{\Theta(s)}{V(s)} = \frac{K}{s\left((Js+b)(Ls+R)+K^2\right)} \quad \left[\frac{rad/sec}{V}\right] \tag{4.26}$$

4.4.1 Otimização do controlador

O controle do sistema configura-se em um problema servo, tendo como entrada uma somatória de senoides conforme a Figura 4.29.

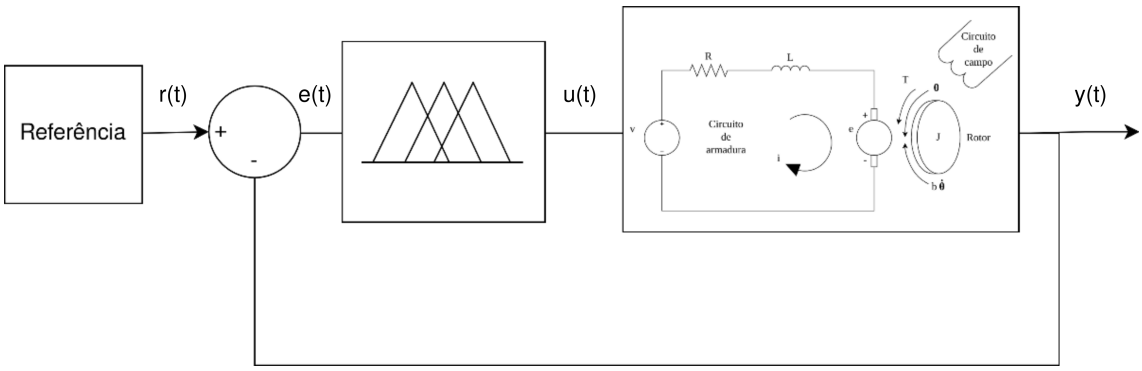
Figura 4.29: Referência para o controle de velocidade do motor DC



Fonte: Autoria própria

O esquema do sistema de controle é apresentado na Figura 4.30.

Figura 4.30: Sistema de controle de velocidade do motor DC



Fonte: Autoria própria

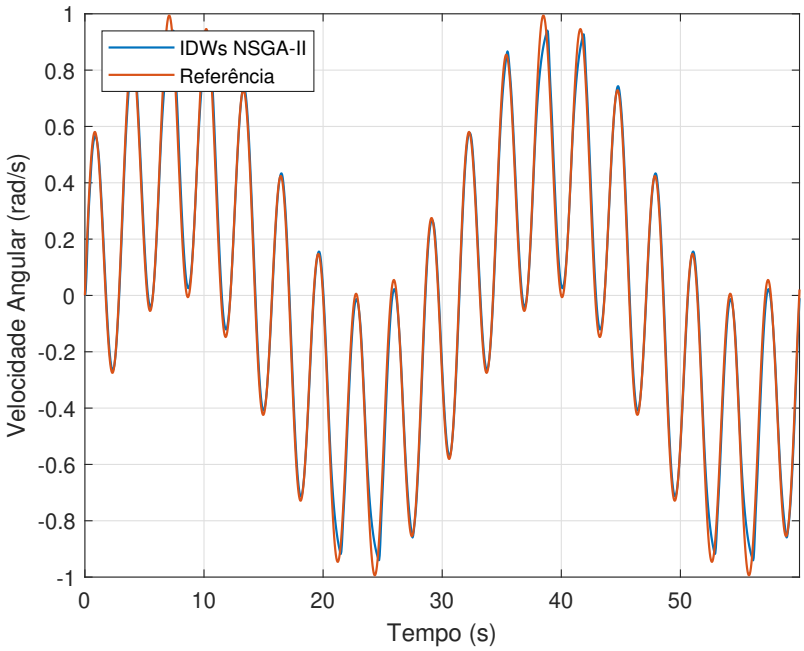
O controlador a ser otimizado é um controlador *fuzzy*-PI Sugeno com três funções de pertinência de forma triangular para cada entrada. Para cada combinação possível entre as funções de pertinência das entradas, há uma regra e sua correspondente saída, totalizando nove funções e saídas. Os valores iniciais dos controladores foram sorteados em torno de valores conhecidos para o problema, a mesma população inicial foi utilizada para todos os algoritmos de otimização, de modo que a aleatoriedade na definição da população inicial não influenciará na comparação das técnicas.

A frequência de amostragem do sistema é de $10^3 Hz$ e foram definidas de forma experimental duas faixas de frequência, a primeira faixa de $0Hz$ a $0,5Hz$ e a segunda de $0,5Hz$ a $500Hz$. Definidas as faixas dos $IDWs$ foi realizada a otimização usando o NSGA-II e NSGA-III. Para ambos os algoritmos de otimização, definiu-se de forma heurística um população de 25 indivíduos, executados por 20 gerações, porcentagem de cruzamento de 80% e a taxa de mutação de 10%. Foram adotados, no NSGA-III, quatro pontos de referências calculados pelo algoritmo. Para a comparação foi realizada a sintonia do controlador pelo GA utilizando como função custo os índices IEA e IEAT.

4.4.2 Resultados

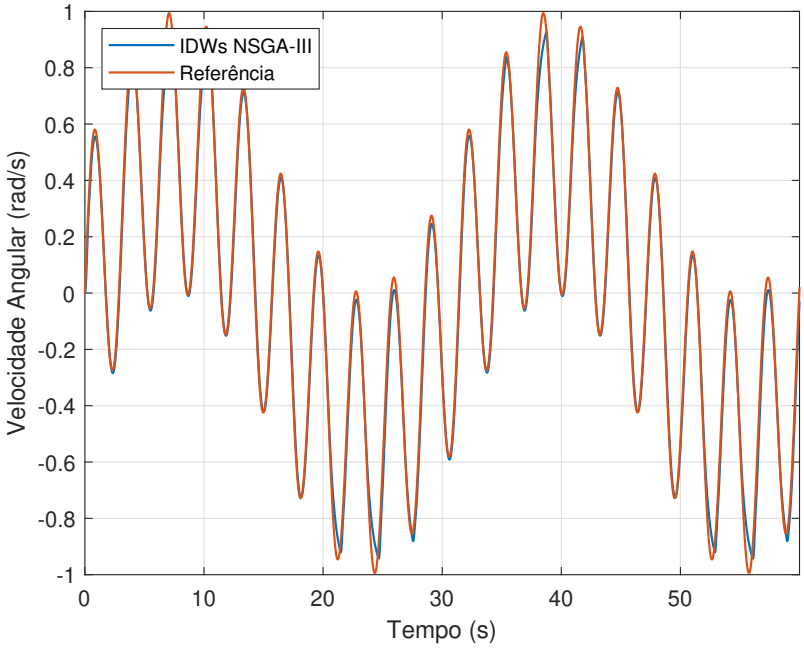
Para a metodologia proposta, aplicada ao controle da velocidade do motor-DC, utilizando a simulação no Matlab, obteve-se para o NSGA-II e o NSGA-III, respectivamente, os resultados expostos na Figura 4.31 e na Figura 4.32.

Figura 4.31: Sinal de resposta referente aos *IDWs* otimizados com NSGA-II para o controle de velocidade do motor DC



Fonte: Autoria própria

Figura 4.32: Sinal de resposta referente aos *IDWs* otimizados com NSGA-III para o controle de velocidade do motor DC

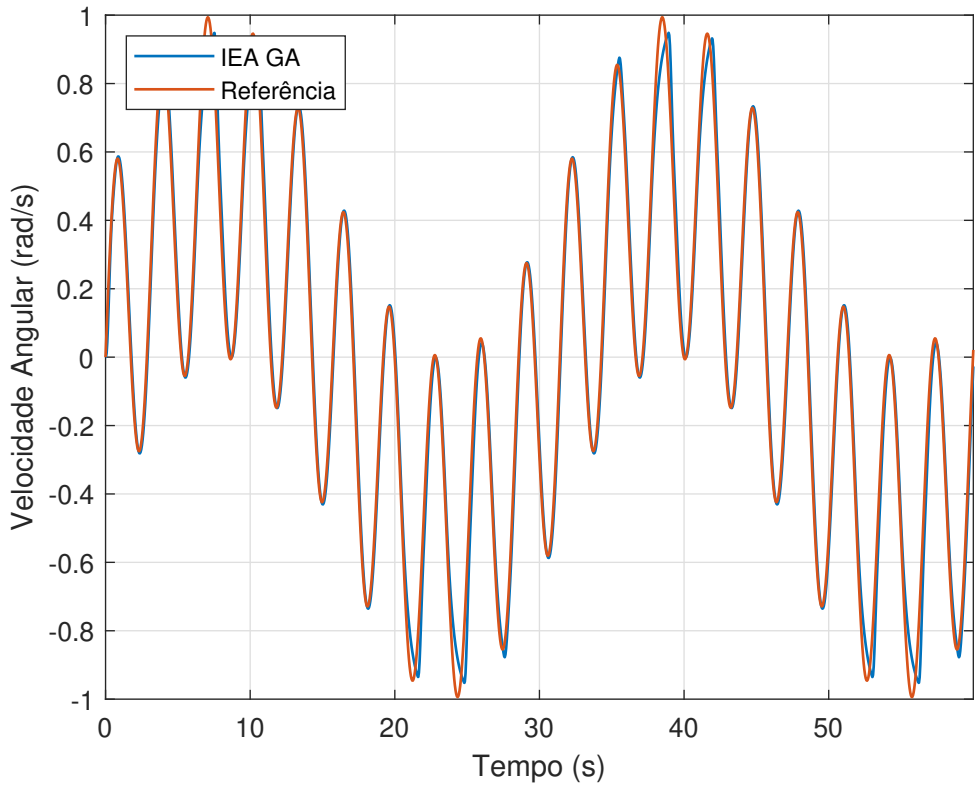


Fonte: Autoria própria

Os controladores encontrados pelo NSGA-II e NSGA-III conseguem seguir bem a referência senoidal.

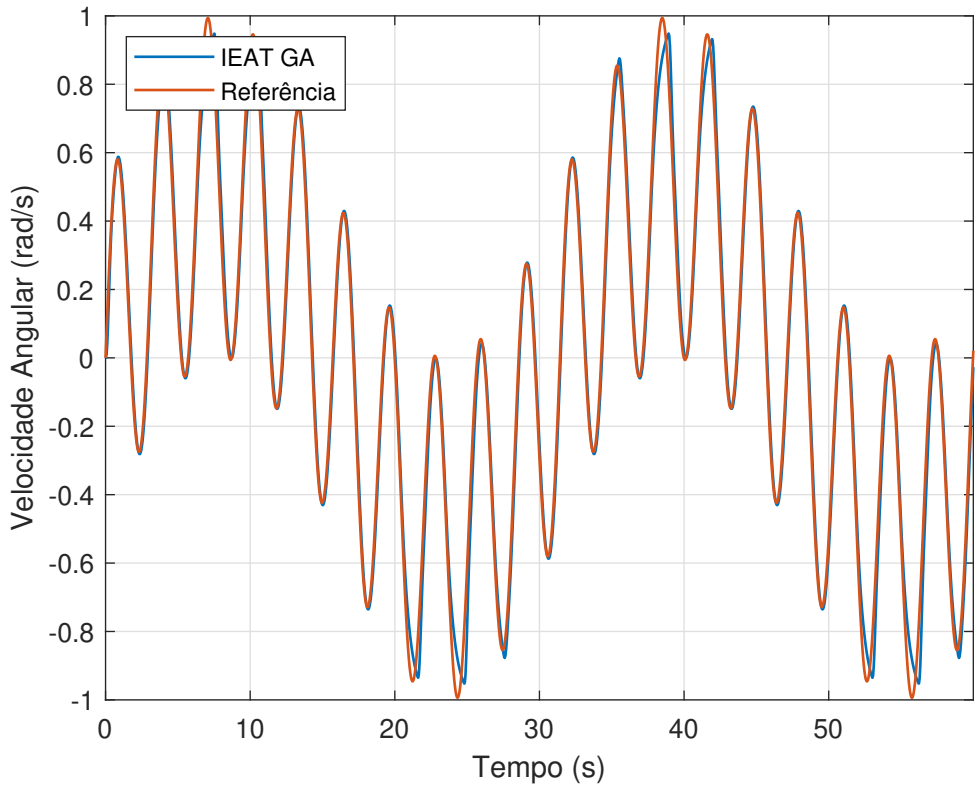
Foi realizada também a otimização de controles utilizando o GA mono-objetivo com o IEA e IEAT como funções custo, para comparação com a metodologia proposta. Obteve-se, então as respostas exibidas na Figura 4.33 para o IEA e na Figura 4.34 para o IEAT.

Figura 4.33: Sinal de resposta referente ao IEA para o controle de velocidade do motor DC



Fonte: Autoria própria

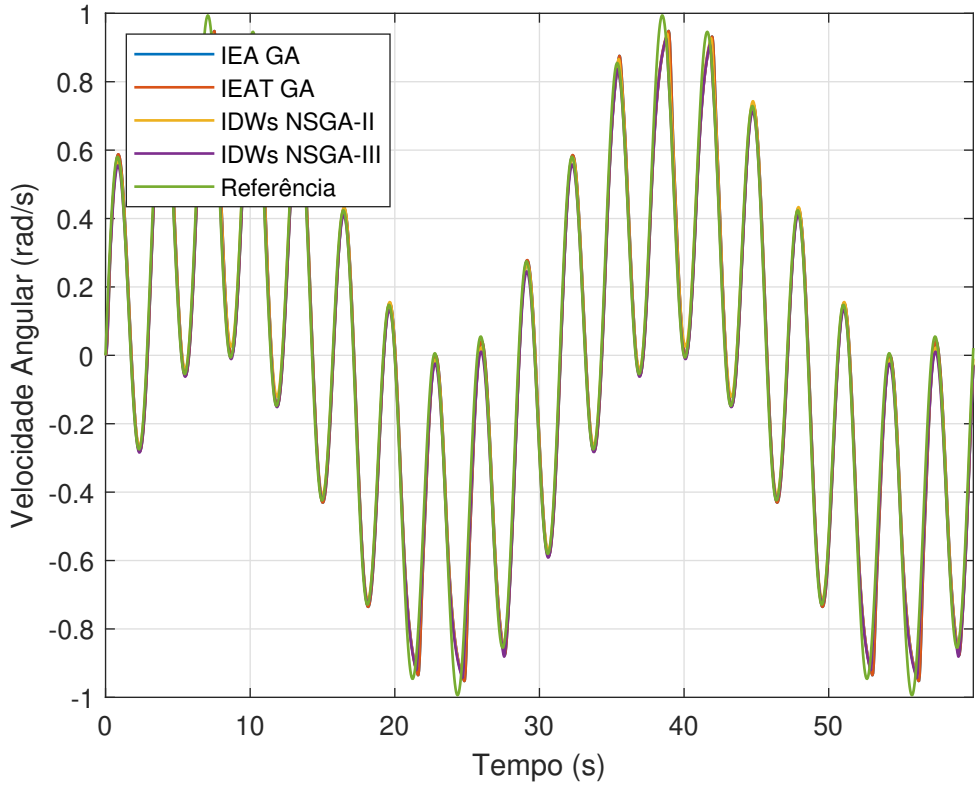
Figura 4.34: Sinal de resposta referente ao IEAT para o controle de velocidade do motor DC



Fonte: Autoria própria

Os controladores encontrados pelo GAusando IEA e IEAT como funções custo, também seguem bem a referência senoidal. A comparação entre os resultados obtidos pelas técnicas está representado na Figura 4.35.

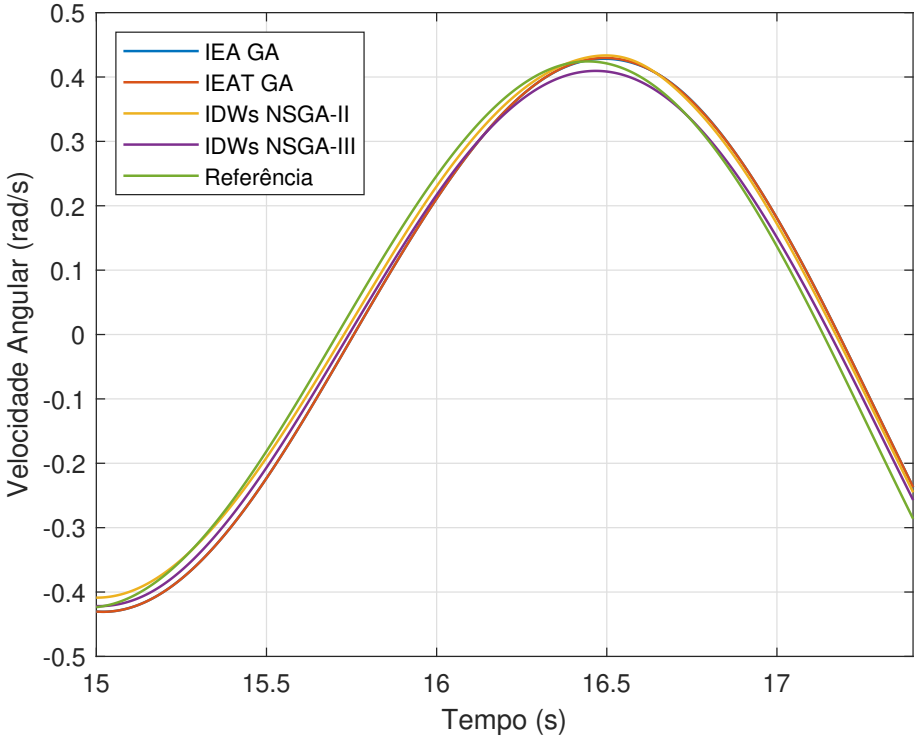
Figura 4.35: Sinais de resposta dos quatro controladores de velocidade do motor DC



Fonte: Autoria própria

Como as respostas apresentadas foram muito próximas, para melhor apresentar o resultado, facilitando a visualização e análise do comportamento das respostas, foi destacado o trecho entre $15,5s$ e $17,3s$, na Figura 4.36.Calculando-se o IEA para cada um dos controladores, o controlador referente ao AG usando IEA alcançou erro absoluto de $24,743rad$, o referente ao AG usando IEAT obteve erro absoluto $24,745rad$, o referente aos $IDWs$ otimizados por NSGA-II apresentou erro absoluto de $24,417ra$, enquanto o referente ao NSGA-III resultou em erro absoluto de $24,162rad$.

Figura 4.36: Recorte da Figura 4.35 para melhor visualização e analise

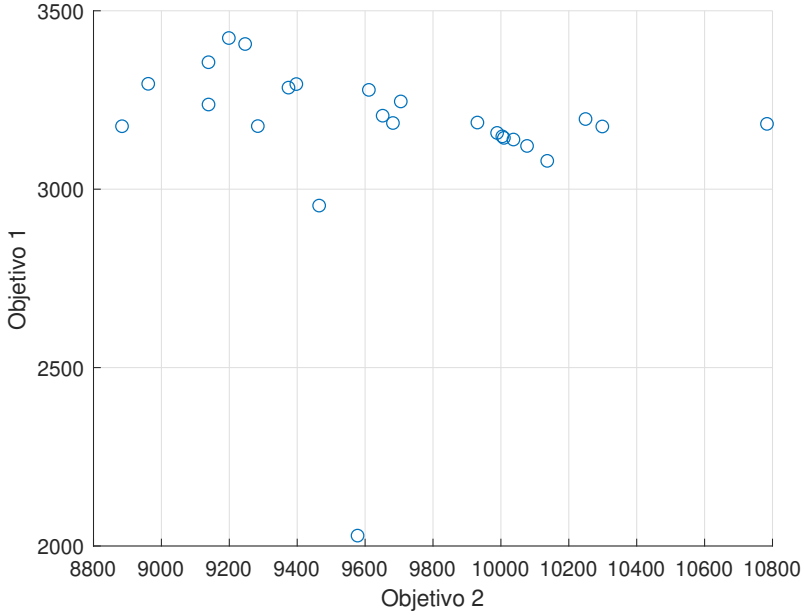


Fonte: Autoria própria

Primeiramente, é importante notar que a resposta para os controladores sintonizados pelo GA usando IEA e IEAT são praticamente idêntica, de modo que, na quase totalidade do tempo, e na totalidade do recorte de tempo exposto na Figura 4.36, a resposta referente ao IEAT sobrepõe a resposta referente ao IEA. Comparando a resposta encontrada com os quatro controladores, observa-se que a resposta referente ao NSGA-II é a que apresenta a resposta mais rápida e agressiva, enquanto a referente ao NSGA-III é a mais suave, apesar de apresentarem comportamentos diversos as respostas apresentadas pelos algoritmos multiobjetivo apresentam um erro absoluto menor que as técnicas mono-objetivo, com vantagem a técnica NSGA-III que, além da maior suavidade apresentou erro absoluto menor.

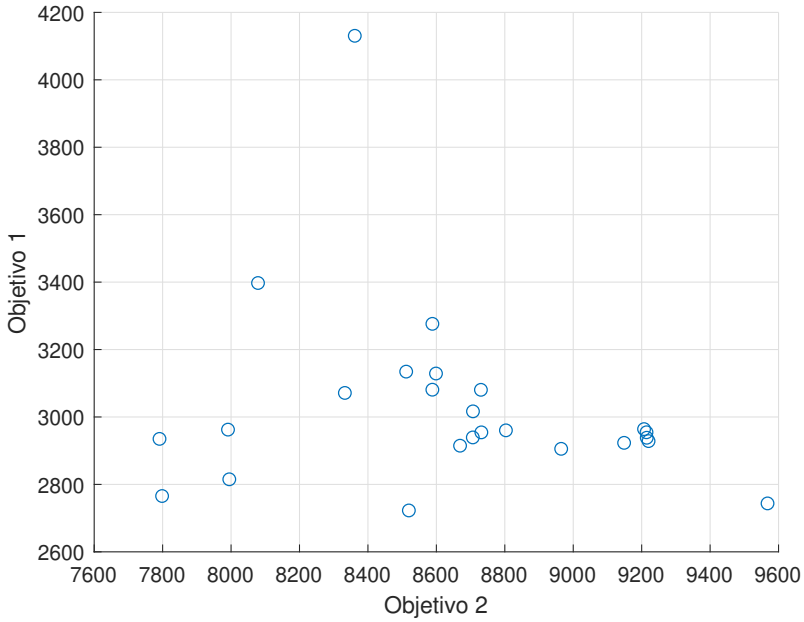
Ademais, as distribuições das populações de soluções encontradas pelo NSGA-II e NSGA-III estão representadas respectivamente nas Figuras 4.37 e 4.38. As populações estão concentradas em uma única região e existem poucas soluções não dominadas, o que se atribui a uma população inicial pouco variada que não representava suficientemente bem o espaço de busca, ou a necessidade de utilização de maiores taxas de mutação para os algoritmos para se obter resultados mais variados.

Figura 4.37: Distribuição da população do NSGA-II quanto aos objetivos



Fonte: Autoria própria

Figura 4.38: Distribuição da população do NSGA-III quanto aos objetivos



Fonte: Autoria própria

Capítulo 5

Conclusão

Diante do apresentado, pode-se observar a viabilidade da utilização da transformada *wavelet*, já muito utilizada nas mais diversas aplicações de análise de sinais, valendo-se dos seus descritores para otimização multiobjetivo de controladores levando em consideração aspectos da resposta do sistema controlado. Durante os testes realizados encontrou-se um bom desempenho da técnica nos estudos de caso, que exemplificam algumas possibilidades que podem ser exploradas usando a metodologia proposta. Observaram-se, porém, algumas limitações quanto a definição das frequências, devido a questões numéricas. Por exemplo, no caso do pêndulo, poderia ter sido observada uma faixa de frequência para o degrau da perturbação, porém, como a alteração do degrau ocorre com a frequência de Nyquist não é possível que essa faixa de frequência seja utilizada, para tanto, seria necessário redefinir o problema.

Pelos resultados obtidos, pode-se observar que a função de avaliação proposta conseguiu mapear adequadamente o desempenho dos controladores para as meta-heurísticas, apontando de forma satisfatória os melhores controladores. A abordagem mostrou-se adequada à análise das oscilações de alta frequência, associadas ao transitório em problemas servo, à análise de rejeição à perturbação em problemas reguladores e em ambos, em problemas servo com perturbação, como observado respectivamente no sistema de tanques acoplados, no sistema pêndulo invertido e no controle de posição do motor DC. Há de se ressaltar ainda, que ao contrário de outras funções de avaliação, a função de avaliação proposta neste trabalho não pressupõe que o sinal de referência seja um degrau, podendo ser aplicada a quaisquer sinais de referência, como demostrado no controle de velocidade do motor DC. Nesse caso de estudo, inclusive, o desempenho encontrado pela técnica proposta, superou a utilização do IEA em seu próprio critério, apontando que o aumento da dimensionalidade do problema em função do erro, pode melhorar a convergência do algoritmo. Além disso, as meta-heurísticas multiobjetivo empregadas devolvem uma população de respostas, podendo o projetista escolher qual controlador irá utilizar de acordo com seu comportamento frente aos objetivos conflitantes.

Nos quatro estudos de caso observou-se um bom desempenho do NSGA-II no domínio *wavelet* frente as outras técnicas. No geral, NSGA-III teve desempenho próximo ao GA utilizando IEA como função custo e aqueles que utilizaram o IEAT, no geral, apresentaram pior desempenho. Se atribui isso a escolha de parâmetros dos algoritmos. Deve-se, nesse contexto, apontar a possível sensibilidade do NSGA-II e do NSGA-III na escolha de seus parâmetros, a uma população pouco numerosa e em uma possível sensibilidade a população

inicial escolhida, em especial no caso do NSGA-III onde uma pequena população sem tanta diversidade e com poucos pontos de referência podem dificultar a seleção por nichamento. Essa dificuldade de manter a população diversa e ao mesmo tempo adaptar a solução ao problema, com os parâmetros adotados, pode ser notada pela distribuição da população em função dos objetivos. Contudo, mesmo assim, no geral as técnicas multiobjetivo adotadas mostraram um bom desempenho, o que se atribui a melhor exploração do espaço de busca pelo aumento de dimensionalidade trazido pela decomposição *wavelet*.

Nesse contexto, a maior exploração dos parâmetros dos algoritmos é uma possibilidade de futuros estudos. Deve-se ressaltar, ainda, que a metodologia proposta pode ser aplicada a uma ampla gama de controladores, não se limitando aos controladores do tipo *fuzzy* e podendo também ser utilizada com diferentes tipos de referências. Assim, trata-se de uma metodologia genérica com um ampla gama de possibilidades de aplicações. Além disso, o uso de outras técnicas de otimização, ou modificações nessas podem trazer algum benefício, visto que as técnicas de otimização até então empregadas são de uso genérico, a utilização de técnicas mais voltadas para esses tipos de problema poderiam trazer algum benefício ao desempenho da otimização, sendo uma possível área de estudos futuros. Outra questão a ser explorada em estudos futuros é a possibilidade de subdividir as métricas em períodos de tempo, por exemplo, separando a avaliação para cada degrau da referência se a referência for dada pela soma de degraus, adequando melhor a otimização a vários pontos de operação do sistema, ou ainda aplicar a ponderação no tempo ao medir a similaridade dos descritores *wavelet*, seguindo lógica semelhante ao cálculo do IEAT.

Referências Bibliográficas

ABEDINIA, O.; AMJADY, N.; GHASEMI, A.; SHAYEGHI, H. Multi-stage fuzzy load frequency control based on multi-objective harmony search algorithm in deregulated environment. *Journal of Operation and Automation in Power Engineering*, University of Mohaghegh Ardabili, v. 1, n. 1, p. 63–73, 2007.

ABRAMOWITZ, M.; STEGUN, I. A. *et al. Handbook of mathematical functions with formulas, graphs, and mathematical tables*. [S.l.]: Dover, New York, 1972. v. 9.

AL-SALAMI, N. M. Evolutionary algorithm definition. *American J. of Engineering and Applied Sciences*, Citeseer, v. 2, n. 4, p. 789–795, 2009.

ALORF, A. A survey of recently developed metaheuristics and their comparative analysis. *Engineering Applications of Artificial Intelligence*, v. 117, p. 105622, 2023. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0952197622006121>.

APKARIAN, J. Coupled water tank experiments manual. *Quanser Consulting Inc., Canada*, 1999.

ARROYO, J. E. C. *et al.* Heurísticas e metaheurísticas para otimização combinatória multiobjetivo. [sn], 2002.

BAZANELLA, A. S.; JUNIOR, J. M. G. da S. *Sistemas de controle: Princípios e métodos de projeto*. [S.l.]: UFRGS, 2005.

BUENO, F. Métodos heurísticos-teoria e implementações. *IFSC. Araranguá*, 2009.

CAMPOS, M. C. M. M. de; TEIXEIRA, H. C. *Controles típicos de equipamentos e processos industriais*. [S.l.]: Edgard Blücher, 2006.

CERVENKA, P.; MOUSTIER, C. D. Sidescan sonar image processing techniques. *IEEE journal of oceanic engineering*, IEEE, v. 18, n. 2, p. 108–122, 1993.

CHAKRADEO, S. S.; HENDRE, A. S.; DESHPANDE, S. U. Generalized theory for hybridization of evolutionary algorithms. In: *2014 IEEE International Conference on Computational Intelligence and Computing Research*. [S.l.: s.n.], 2014. p. 1–5.

CHANKONG, V.; HAIMES, Y. Y. Multiobjective decision making: theory and methodology. In: *North Holland series in system science and engineering*. [S.l.]: North-Holland, 1983.

CHEN, C.-T. *Linear system theory and design*. [S.l.]: Oxford University Press, Inc., 1995.

CHEN, P.; ZHANG, W. Improvement on an inverted decoupling technique for a class of stable linear multivariable processes. *ISA transactions*, Elsevier, v. 46, n. 2, p. 199–210, 2007.

CIVICIOGLU, P.; BESDOK, E. A conceptual comparison of the cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artificial Intelligence Review*, Springer, v. 39, n. 4, p. 315–346, 2013.

COELHO, L. d. S.; MARIANI, V. C. Firefly algorithm approach based on chaotic tinker-bell map applied to multivariable pid controller tuning. *Computers & Mathematics with Applications*, Elsevier, v. 64, n. 8, p. 2371–2382, 2012.

COELLO, C. A. An updated survey of ga-based multiobjective optimization techniques. *ACM Computing Surveys (CSUR)*, ACM, v. 32, n. 2, p. 109–143, 2000.

COELLO, C. A. C. An introduction to evolutionary algorithms and their applications. In: SPRINGER. *Advanced Distributed Systems: 5th International School and Symposium, ISSADS 2005, Guadalajara, Mexico, January 24-28, 2005, Revised Selected Papers 5*. [S.l.], 2005. p. 425–442.

COELLO, C. C.; LECHUGA, M. Mopso: a proposal for multiple objective particle swarm optimization. In: *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*. [S.l.: s.n.], 2002. v. 2, p. 1051–1056 vol.2.

COHON, J. L. *Multiobjective programming and planning*. [S.l.]: Courier Corporation, 2004. v. 140.

CUNHA R. TAKAHASHI, C. H. A. A. G. Manual de computação evolutiva e metaheurística. Imprensa da Universidade de Coimbra, Coimbra, 2012. Disponível em: <https://digitalis.uc.pt/handle/10316.2/5655>.

CZYZŻAK, P.; JASZKIEWICZ, A. Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, Wiley Online Library, v. 7, n. 1, p. 34–47, 1998.

DANIELS, R. L. Analytical evaluation of multi-criteria heuristics. *Management Science*, INFORMS, v. 38, n. 4, p. 501–513, 1992.

DANIELS, R. L.; CHAMBERS, R. J. Multiobjective flow-shop scheduling. *Naval Research Logistics (NRL)*, Wiley Online Library, v. 37, n. 6, p. 981–995, 1990.

DARILMAZ, I. *Wavelet based similarity measurement algorithm for seafloor morphology*. Tese (Doutorado) — Massachusetts Institute of Technology, 2006.

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.

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 Transactions on Evolutionary Computation*, v. 18, n. 4, p. 577–601, 2014.

DEB, K.; PRATAP, A.; AGARWAL, S.; MEYARIVAN, T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, IEEE, v. 6, n. 2, p. 182–197, 2002.

DIMEO, R.; LEE, K. Y. Boiler-turbine control system design using a genetic algorithm. *IEEE transactions on energy conversion*, IEEE, v. 10, n. 4, p. 752–759, 1995.

DO, M. N.; VETTERLI, M. Wavelet-based texture retrieval using generalized gaussian density and kullback-leibler distance. *IEEE transactions on image processing*, IEEE, v. 11, n. 2, p. 146–158, 2002.

DOMAŃSKI, P. D. *et al. Control Performance Assessment: Theoretical Analyses and Industrial Practice*. [S.l.]: Springer, 2020. v. 245.

DORF, R. C.; BISHOP, R. H.; CANTO, S. D.; CANTO, R. D.; DORMIDO, S. *Sistemas de control moderno*. [S.l.]: Pearson Prentice Hall, 2005.

DORIGO, M.; STÜTZLE, T. Ant colony optimization algorithms for the traveling salesman problem. MIT Press, 2004.

EBERHART, R. C.; KENNEDY, J. *et al.* A new optimizer using particle swarm theory. In: NEW YORK, NY. *Proceedings of the sixth international symposium on micro machine and human science*. [S.l.], 1995. v. 1, p. 39–43.

EHRGOTT, M.; GANDIBLEUX, X. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR-Spektrum*, Springer, v. 22, n. 4, p. 425–460, 2000.

EIBEN, A. E.; SMITH, J. E. *Introduction to evolutionary computing*. [S.l.]: Springer, 2015.

FOGEL, L. J. *ON THE DESIGN OF CONSCIOUS AUTOMATA*. [S.l.], 1966.

FONSECA, C. M.; FLEMING, P. J. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, v. 3, n. 1, p. 1–16, 1995.

FONSECA, C. M.; FLEMING, P. J. *et al.* Genetic algorithms for multiobjective optimization: formulationdiscussion and generalization. In: CITESEER. *Icga*. [S.l.], 1993. v. 93, n. July, p. 416–423.

FOURMAN, M. P. Compaction of symbolic layout using genetic algorithms. In: PSYCHOLOGY PRESS. *Proceedings of the first international conference on genetic algorithms and their applications*. [S.l.], 2014. p. 141–153.

GAGNON, E.; POMERLEAU, A.; DESBIENS, A. Simplified, ideal or inverted decoupling? *ISA transactions*, Elsevier, v. 37, n. 4, p. 265–276, 1998.

GARRIDO, J.; VÁZQUEZ, F.; MORILLA, F. An extended approach of inverted decoupling. *Journal of Process Control*, Elsevier, v. 21, n. 1, p. 55–68, 2011.

GLOVER, F.; GREENBERG, H. J. New approaches for heuristic search: A bilateral linkage with artificial intelligence. *European Journal of Operational Research*, Elsevier, v. 39, n. 2, p. 119–130, 1989.

GLOVER, F.; LAGUNA, M. Tabu search. *Kluwer Academic Publishers*, 1997.

GOLDBERG, D. E. Genetic algorithms in search, optimization and machine learning addison welssey publishing company. *Reading, MA*, 1989.

GOLDBERG, D. E.; RICHARDSON, J. *et al.* Genetic algorithms with sharing for multimodal function optimization. In: HILLSDALE, NJ: LAWRENCE ERLBAUM. *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*. [S.l.], 1987. v. 4149.

GOODHART, S.; BURNHAM, K.; JAMES, D. Bilinear self-tuning control of a high temperature heat treatment plant. *IEE Proceedings-Control Theory and Applications*, IET, v. 141, n. 1, p. 12–18, 1994.

GOSMANN, H. L. *Um sistema Multivariável de tanques acoplados para avaliação de técnicas de controle*. Tese (Doutorado) — Dissertação de Mestrado, Universidade de Brasília, 2002.

GREENWOOD, G. W.; HU, X.; D'AMBROSIO, J. G. Fitness functions for multiple objective optimization problems: Combining preferences with pareto rankings. In: *Foga*. [S.l.: s.n.], 1996. v. 96, p. 437–455.

GUPTA, S.; BISWAS, P. K.; DEBNATH, S.; GHOSH, A.; BABU, T. S.; ZAWBAA, H. M.; KAMEL, S. Metaheuristic optimization techniques used in controlling of an active magnetic bearing system for high-speed machining application. *IEEE Access*, v. 11, p. 12100–12118, 2023.

HAJELA, P.; LIN, C. Y. Genetic search strategies in multicriterion optimal design. *Structural optimization*, Springer, v. 4, p. 99–107, 1992.

HANSEN, M. P.; JASZKIEWICZ, A. *Evaluating the quality of approximations to the non-dominated set*. [S.l.]: IMM, Department of Mathematical Modelling, Technical Universityof Denmark, 1994.

HASHIMOTO, K. *Técnicas de otimização combinatória multiobjetivo aplicadas na estimação do desempenho elétrico de redes de distribuição.* Tese (Doutorado) — Universidade de São Paulo, 2004.

HEDAYATZADEH, R.; HASANIZADEH, B.; AKBARI, R.; ZIARATI, K. A multi-objective artificial bee colony for optimizing multi-objective problems. In: *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*. [S.l.: s.n.], 2010. v. 5, p. V5–277–V5–281.

HOLLAND, J. H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* [S.l.]: U Michigan Press, 1975.

HORN, J.; NAFPLIOTIS, N.; GOLDBERG, D. A niched pareto genetic algorithm for multiobjective optimization. In: *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*. [S.l.: s.n.], 1994. p. 82–87 vol.1.

HORN, J. rey; NAFPLIOTIS, N.; GOLDBERG, D. E. Multiobjective optimization using the niched pareto genetic algorithm. *IlliGAL report*, v. 93005, p. 61801–2296, 1993.

ISHIBUCHI, H.; MURATA, T. Multi-objective genetic local search algorithm. In: IEEE. *Proceedings of IEEE international conference on evolutionary computation*. [S.l.], 1996. p. 119–124.

ISSA, M. Enhanced arithmetic optimization algorithm for parameter estimation of pid controller. *Arabian Journal for Science and Engineering*, Springer, v. 48, n. 2, p. 2191–2205, 2023.

JOHANSSON, K. H.; HORCH, A.; WIJK, O.; HANSSON, A. Teaching multivariable control using the quadruple-tank process. In: IEEE. *Decision and Control, 1999. Proceedings of the 38th IEEE Conference on*. [S.l.], 1999. v. 1, p. 807–812.

JONES, D. F.; MIRRAZAVI, S. K.; TAMIZ, M. Multi-objective meta-heuristics: An overview of the current state-of-the-art. *European journal of operational research*, Elsevier, v. 137, n. 1, p. 1–9, 2002.

JONG, K. D.; FOGEL, D. B.; SCHWEFEL, H.-P. A2. 3 a history of evolutionary computation. *A1. 1 Introduction*, Citeseer, 1997.

KAGAN, N. *Electrical power distribution systems planning using multiobjective and fuzzy mathematical programming*. Tese (Doutorado), 1993.

KARABOGA, D. Artificial bee colony algorithm. *scholarpedia*, v. 5, n. 3, p. 6915, 2010.

KARABOGA, D.; BASTURK, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of global optimization*, Springer, v. 39, p. 459–471, 2007.

KAVANAGH, R. Noninteracting controls in linear multivariable systems. *Transactions of the American Institute of Electrical Engineers, Part II: Applications and Industry*, IEEE, v. 76, n. 2, p. 95–100, 1957.

KE, L. Damage detection method of long-span bridge cable structure based on wavelet packet analysis. In: IEEE. *2022 14th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*. [S.l.], 2022. p. 326–330.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: *Proceedings of ICNN'95 - International Conference on Neural Networks*. [S.l.: s.n.], 1995. v. 4, p. 1942–1948 vol.4.

KIRKPATRICK, S.; JR, C. D. G.; VECCHI, M. P. Optimization by simulated annealing. *science*, American association for the advancement of science, v. 220, n. 4598, p. 671–680, 1983.

KOKARE, M.; CHATTERJI, B.; BISWAS, P. Comparison of similarity metrics for texture image retrieval. In: IEEE. *TENCON 2003. Conference on Convergent Technologies for the Asia-Pacific Region*. [S.l.], 2003. v. 2, p. 571–575.

KOZA, J. R. *et al.* Evolution of subsumption using genetic programming. In: MIT PRESS CAMBRIDGE, MA, USA. *Proceedings of the first European conference on artificial life*. [S.l.], 1992. p. 110–119.

KRONLAND-MARTINET, R.; MORLET, J.; GROSSMANN, A. Analysis of sound patterns through wavelet transforms. *International journal of pattern recognition and artificial intelligence*, World Scientific, v. 1, n. 02, p. 273–302, 1987.

KUKHARCHUK, V. V.; KAZYV, S. S.; BYKOVSKY, S. A.; WÓJCIK, W.; KOTYRA, A.; AKHMETOVA, A.; BAZAROVA, M. Discrete wavelet transformation in spectral analysis of vibration processes at hydropower units. *Przeglad elektrotechniczny*, Wydawnictwo SIGMA, v. 93, n. 5, p. 65–68, 2017.

KURSAWE, F. Evolution strategies for vector optimization. In: CITESEER. *Preliminary Proceedings of the 10th International Conference on Multiple Criteria Decision Making*. [S.l.], 1992. p. 187–193.

LAINE, A.; FAN, J. Texture classification by wavelet packet signatures. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, v. 15, n. 11, p. 1186–1191, 1993.

LIN, J. Feature extraction of machine sound using wavelet and its application in fault diagnosis. *NDT & e International*, Elsevier, v. 34, n. 1, p. 25–30, 2001.

LIU, J.; CHI, Y.; ZHU, C. A dynamic multiagent genetic algorithm for gene regulatory network reconstruction based on fuzzy cognitive maps. *IEEE Transactions on Fuzzy Systems*, IEEE, v. 24, n. 2, p. 419–431, 2015.

LUYBEN, W. L. Distillation decoupling. *AIChE Journal*, Wiley Online Library, v. 16, n. 2, p. 198–203, 1970.

MAGALHÃES, H. d. O. *Análise de sinais para engenheiros: Uma abordagem via Wavelet*. [S.l.]: Rio de Janeiro: Brasport, 2007.

MAHFOUD, S. Niching methods for genetic algorithms (ph. d. thesis, university of illinois at urbana-champaign). *Available as IlliGAL Report*, n. 95001, 1995.

MALLAT, S. *A wavelet tour of signal processing*. [S.l.]: Academic press, 1999.

MALLAT, S. G. A theory for multiresolution signal decomposition: the wavelet representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Ieee, v. 11, n. 7, p. 674–693, 1989.

MAMDANI, E. H.; ASSILIAN, S. An experiment in linguistic synthesis with a fuzzy logic controller. *International journal of man-machine studies*, Elsevier, v. 7, n. 1, p. 1–13, 1975.

MANJUNATH, B. S.; MA, W.-Y. Texture features for browsing and retrieval of image data. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, v. 18, n. 8, p. 837–842, 1996.

MARLER, R. T.; ARORA, J. S. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, Springer, v. 26, n. 6, p. 369–395, 2004.

MARLIN, T. E. *Process Control*. [S.l.]: New York: McGraw-Hill, 1995.

MIRJALILI, S.; LEWIS, A. The whale optimization algorithm. *Advances in Engineering Software*, Elsevier, v. 95, p. 51–67, 2016.

MIRJALILI, S.; MIRJALILI, S. M.; LEWIS, A. Grey wolf optimizer. *Advances in engineering software*, Elsevier, v. 69, p. 46–61, 2014.

MIRJALILI, S.; MIRJALILI, S. M.; LEWIS, A. Grey wolf optimizer. *Advances in engineering software*, Elsevier, v. 69, p. 46–61, 2014.

MIRJALILI, S.; SAREMI, S.; MIRJALILI, S. M.; COELHO, L. dos S. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Systems with Applications*, v. 47, p. 106–119, 2016. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417415007435>.

ONOFRE, M. d. P. Lógica fuzzy para controle de ph em um processo petrolífero. Universidade Federal do Rio Grande do Norte, 2011.

OZUMCAN, S.; OZTURK, A.; VARAN, M.; ANDIC, C. A novel honey badger algorithm based load frequency controller design of a two-area system with renewable energy sources. *Energy Reports*, Elsevier, v. 9, p. 272–279, 2023.

PANT, S.; NEMA, R.; GUPTA, S. Detecting faults in power transformers using wavelet transform. In: IEEE. *2021 IEEE 2nd International Conference On Electrical Power and Energy Systems (ICEPES)*. [S.l.], 2021. p. 1–5.

RAMOS, D. B.; FARRET, F. A.; LENZ, J. M.; FERRIGOLO, F. Z. Uma metodologia experimental para modelagem e controle da eficiência de células a combustível de membrana de troca protônica.

RASHEDI, E.; NEZAMABADI-POUR, H.; SARYAZDI, S. Gsa: a gravitational search algorithm. *Information sciences*, Elsevier, v. 179, n. 13, p. 2232–2248, 2009.

RECHENBERG, I. *Evolutionary strategies: optimizing technical systems with principles of biological evolution*. [S.l.]: Frommann-Holzboog Verlag: Stuttgart, 1973.

RINGUEST, J. L. *Multiobjective optimization: behavioral and computational considerations*. [S.l.]: Springer Science & Business Media, 2012.

SAMPAT, M. P.; WANG, Z.; GUPTA, S.; BOVIK, A. C.; MARKEY, M. K. Complex wavelet structural similarity: A new image similarity index. *IEEE transactions on image processing*, IEEE, v. 18, n. 11, p. 2385–2401, 2009.

SANTOS, F. B. B. Implementação eficiente de busca em plataforma paralela. Congresso da Sociedade Brasileira de Computação, Florianópolis, 2002.

SANTOS, J. E. S. D. *et al.* Controle preditivo não-linear para sistemas de hammerstein. Florianópolis, SC, 2007.

SAXENA, A.; KUMAR, J.; DEOLIA, V. K.; SHARMA, K. Npid controller gain optimization for two link manipulator system with payload. In: AIP PUBLISHING. *AIP Conference Proceedings*. [S.l.], 2023. v. 2721, n. 1.

SCHAFFER, J. D. *Multiple objective optimization with vector evaluated genetic algorithms*. Tese (Doutorado) — Vanderbilt University, Nashville, TN, 1984.

SEBORG, D. E.; MELLICHAMP, D. A.; EDGAR, T. F.; III, F. J. D. *Process dynamics and control*. [S.l.]: John Wiley & Sons, 2010.

SHEN, X.; CHEN, J.-G.; ZHU, X.-C.; LIU, P.-Y.; DU, Z.-H. Multi-objective optimization of wind turbine blades using lifting surface method. *Energy*, v. 90, p. 1111–1121, 2015. ISSN 0360-5442. Disponível em: <https://www.sciencedirect.com/science/article/pii/S036054421500818X>.

SHINSKEY, F. *Process-control systems: application, design, adjustment*. McGraw-Hill, 1988. (Chemical engineering series). ISBN 9780070569034. Disponível em: <https://books.google.com.br/books?id=PE1lAg3ks0YC>.

SIMÕES, M. G.; SHAW, I. S. Controle e modelagem fuzzy. *São Paulo. Blucher: Fapesp*, 2007.

SMITH, C. A.; CORRIPIO, A. B. *Principles and practice of automatic process control*. [S.l.]: Wiley New York, 1985. v. 2.

SOCHA, K.; DORIGO, M. Ant colony optimization for continuous domains. *European journal of operational research*, Elsevier, v. 185, n. 3, p. 1155–1173, 2008.

SORNSEN, I.; SUPPITAKSAKUL, C.; KITPAIBOONTAWEE, R. Partial discharge signal detection in generators using wavelet transforms. In: IEEE. *2021 International Conference on Power, Energy and Innovations (ICPEI)*. [S.l.], 2021. p. 195–198.

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

STEUER, R. E.; STEUER, R. *Multiple criteria optimization: theory, computation, and application*. [S.l.]: Wiley New York, 1986. v. 233.

STORN, R.; PRICE, K. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, Springer Nature BV, v. 11, n. 4, p. 341, 1997.

STRANG, G.; NGUYEN, T. *Wavelets and filter banks*. [S.l.]: SIAM, 1996.

SUN, C.; ZHOU, H.; CHEN, L. Improved differential evolution algorithms. In: *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*. [S.l.: s.n.], 2012. v. 3, p. 142–145.

TAKAGI, T.; SUGENO, M. Fuzzy identification of systems and its applications to modeling and control. *Systems, Man and Cybernetics, IEEE Transactions on*, IEEE, n. 1, p. 116–132, 1985.

TANG, K.-S.; MAN, K. F.; CHEN, G.; KWONG, S. An optimal fuzzy pid controller. *IEEE Transactions on Industrial Electronics*, IEEE, v. 48, n. 4, p. 757–765, 2001.

TIAN, X.; LI, J. Robust aerodynamic shape optimization using a novel multi-objective evolutionary algorithm coupled with surrogate model. *Structural and Multidisciplinary Optimization*, Springer, v. 62, p. 1969–1987, 2020.

TILBURY, D.; LUNTZ, J.; MESSNER, W. *Control Tutorials for MATLAB and Simulink*. 1998. Disponível em: <https://ctms.engin.umich.edu/CTMS/>.

ULUNGU, E.; TEGHEM, J.; OST, C. Efficiency of interactive multi-objective simulated annealing through a case study. *Journal of the Operational Research Society*, Taylor & Francis, v. 49, n. 10, p. 1044–1050, 1998.

VELDHUIZEN, D. A. V.; LAMONT, G. B. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary computation*, MITP, v. 8, n. 2, p. 125–147, 2000a.

VELDHUIZEN, D. A. V.; LAMONT, G. B. On measuring multiobjective evolutionary algorithm performance. In: IEEE. *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512)*. [S.l.], 2000b. v. 1, p. 204–211.

VIKHAR, P. A. Evolutionary algorithms: A critical review and its future prospects. In: IEEE. *2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC)*. [S.l.], 2016. p. 261–265.

VILANI, M. T.; SANCHES, L. Análise de fourier e wavelets aplicada à temperatura do ar em diferentes tipologias de ocupação. *Revista Brasileira de Engenharia Agrícola e Ambiental*, SciELO Brasil, v. 17, p. 1340–1346, 2013.

WADE, H. L. Inverted decoupling: a neglected technique. *ISA transactions*, Elsevier, v. 36, n. 1, p. 3–10, 1997.

WANG, J. Z.; WIEDERHOLD, G.; FIRSCHEIN, O.; WEI, S. X. Wavelet-based image indexing techniques with partial sketch retrieval capability. In: IEEE. *Digital Libraries, 1997. ADL'97. Proceedings., IEEE International Forum on Research and Technology Advances in*. [S.l.], 1997. p. 13–24.

WANG, Q.-G. Autotuning of pid controllers.: Cc yu. Elsevier, 2001.

WANG, Z.; BOVIK, A. C.; SHEIKH, H. R.; SIMONCELLI, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, IEEE, v. 13, n. 4, p. 600–612, 2004.

WANG, Z.; SIMONCELLI, E. P. Translation insensitive image similarity in complex wavelet domain. In: IEEE. *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on*. [S.l.], 2005. v. 2, p. ii–573.

WHITLEY, D. An overview of evolutionary algorithms: practical issues and common pitfalls. *Information and software technology*, Elsevier, v. 43, n. 14, p. 817–831, 2001.

WITHERIDGE, S.; PASSOW, B. N.; SHELL, J. Logan's run: Lane optimisation using genetic algorithms based on nsga-ii. In: *2014 International Joint Conference on Neural Networks (IJCNN)*. [S.l.: s.n.], 2014. p. 63–68.

XUE, F.; SANDERSON, A.; GRAVES, R. Multi-objective differential evolution and its application to enterprise planning. In: *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*. [S.l.: s.n.], 2003. v. 3, p. 3535–3541 vol.3.

YANG, X.-S. Firefly algorithms for multimodal optimization. In: SPRINGER. *International symposium on stochastic algorithms*. [S.l.], 2009. p. 169–178.

YANG, X.-S.; DEB, S. Cuckoo search via lévy flights. In: IEEE. *2009 World congress on nature & biologically inspired computing (NaBIC)*. [S.l.], 2009. p. 210–214.

YANG, Y.; CAO, L.; WANG, C.; ZHOU, Q.; JIANG, P. Multi-objective process parameters optimization of hot-wire laser welding using ensemble of metamodels and nsga-ii. *Robotics and Computer-Integrated Manufacturing*, v. 53, p. 141–152, 2018. ISSN 0736-5845. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0736584517303162>.

YAO, X. Global optimisation by evolutionary algorithms. In: *Proceedings of IEEE International Symposium on Parallel Algorithms Architecture Synthesis*. [S.l.: s.n.], 1997. p. 282–291.

ZADEH, L. A. Fuzzy sets. *Information and control*, Elsevier, v. 8, n. 3, p. 338–353, 1965.

ZARACHOFF, M. M.; SHEIKH-AKBARI, A.; MONEKOSSO, D. Non-decimated wavelet based multi-band ear recognition using principal component analysis. *IEEE Access*, IEEE, v. 10, p. 3949–3961, 2021.

ZHANG, L.; ZHANG, L.; MOU, X.; ZHANG, D. Fsim: A feature similarity index for image quality assessment. *IEEE transactions on Image Processing*, IEEE, v. 20, n. 8, p. 2378–2386, 2011.

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.

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

ZITZLER, E.; THIELE, L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, IEEE, v. 3, n. 4, p. 257–271, 1999.