

UNIVERSIDADE FEDERAL DE MINAS GERAIS  
Escola de Engenharia  
Programa de Pós-graduação em Engenharia Elétrica

Paulo Pinheiro Junqueira

**Algoritmo Evolutivo Multi-objetivo  
baseado em Decomposição com  
Arquivo Externo e Adaptação de  
Pesos baseada em Vizinhaça Local**

Belo Horizonte

2021

Paulo Pinheiro Junqueira

**Algoritmo Evolutivo Multi-objetivo baseado em  
Decomposição com Arquivo Externo e Adaptação de  
Pesos baseada em Vizinhança Local**

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do Título de Mestre em Engenharia Elétrica.

UNIVERSIDADE FEDERAL DE MINAS GERAIS

Escola de Engenharia

Programa de Pós-graduação em Engenharia Elétrica

Orientador: Prof. Dr. Frederico Gadelha Guimarães

Coorientador: Prof. Dr. Ivan Reinaldo Meneghini

Belo Horizonte

2021

J95a

Junqueira, Paulo Pinheiro.

Algoritmo evolutivo multi-objetivo baseado em decomposição com arquivo externo e adaptação de pesos baseada em vizinhança local [recurso eletrônico] / Paulo Pinheiro Junqueira. - 2021.

1 recurso online (117 f. : il., color.) : pdf.

Orientador: Frederico Gadelha Guimarães.

Coorientador: Ivan Reinaldo Meneghini.

Dissertação (mestrado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Apêndices: f. 101-117.

Inclui bibliografia.

Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Otimização multiobjectivo - Teses. 3. Método de decomposição - Teses. 4. Algoritmos evolutivos - Teses. I. Guimarães, Frederico Gadelha. II. Meneghini, Ivan Reinaldo. III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU: 621.3(043)

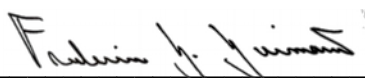
**"Algoritmo Evolutivo Multi-objetivo Baseado Em  
Decomposição Com Arquivo Externo e Adaptação de Pesos  
Baseada Em Vizinhança Local"**

**Paulo Pinheiro Junqueira**

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Mestre em Engenharia Elétrica.

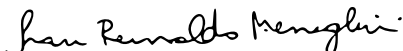
Aprovada em 07 de dezembro de 2021.

Por:



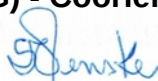
---

**Prof. Dr. Frederico Gadelha Guimarães  
DEE (UFMG) - Orientador**



---

**Prof. Dr. Ivan Reinaldo Meneghini  
(IFMG) - Coorientador**



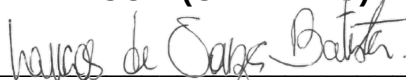
---

**Prof. Dr. Sandra Mara Guse Scos Venske  
Departamento de Ciência da Computação (Unicentro Paraná)**



---

**Prof. Dr. Elizabeth Fialho Wanner  
DECOM (CEFET-MG)**



---

**Prof. Dr. Lucas de Souza Batista  
DEE (UFMG)**

*Dedico mais uma vez aos meus pais, Vera e Paulo. Nada do que conquistei até hoje seria possível se não fosse por eles.*

# Agradecimentos

Primeiramente gostaria de agradecer ao suporte de toda a minha família que participou direta ou indiretamente. Em especial, gostaria de agradecer aos meus pais, Vera e Paulo que desde o início me apoiaram em todas as etapas, sempre incentivando a buscar pelo conhecimento e me aprimorar cada vez mais. Sem eles nenhuma conquista seria possível. Palavra nenhuma é suficiente para descrever a gratidão que sinto, deixo registrado o meu muito obrigado por tudo!

Agradeço também aos meus orientadores, Frederico Gadelha Guimarães e Ivan Reinaldo Meneghini, pela oportunidade, tempo, dedicação, paciência e todos os ensinamentos ao longo deste período e trabalho. Sem dúvidas, vocês são uma grande inspiração e exemplo de pesquisador, profissional e de pessoa que um dia quero me tornar.

Aos meus amigos, os professores e servidores da UFMG, muito obrigado por me ajudarem a enfrentar todas as etapas e por tornarem os dias mais leves ao longo desta caminhada.

Muito obrigado!

*“Talento é 1% inspiração e 99% transpiração.”*

Thomas Edison

# Resumo

Algoritmos evolucionários multi-objetivo, *Multiobjective Evolutionary Algorithm* (MOEA), apresentam uma abordagem interessante para resolver diversos tipos de problemas, conhecidos como problemas multi-objetivo, *Multiobjective Problem* (MOP). A subcategoria de MOEA com abordagens baseada em decomposição vêm crescendo rapidamente e muitos estudos têm demonstrado que a distribuição de vetores de peso desempenha um fator interessante para ajudar a obter um conjunto uniforme de soluções. No entanto, uma distribuição uniforme dos vetores de peso no início da evolução nem sempre resulta em um conjunto uniforme de soluções no espaço de objetivos, pois os resultados se mostram dependentes do formato da fronteira Pareto. Fronteiras Pareto com forma irregular (desconectadas, invertidas, etc.), geralmente não estão presentes em todas as partes do conjunto inicial de vetores de peso. Uma abordagem para superar esse problema é adaptar os vetores de peso buscando aproximar o formato da fronteira Pareto. Visando contribuir com o campo de estudo, é proposto um algoritmo baseado em decomposição que adapta progressivamente seus vetores de peso durante o processo de evolução utilizando um arquivo externo de soluções não-dominadas. O algoritmo proposto é denominado de *Multi-objective Evolutionary Algorithm based on Decomposition with Local-Neighborhood Adaptation* (MOEA/D-LNA). Posteriormente, o algoritmo proposto é comparado com outros algoritmos da literatura em três conjuntos de funções de teste, DTLZ, WFG, MaF e o também resultante desta pesquisa *Generalized Position-Distance* (GPD), com diferentes procedimentos de inicialização de vetores de peso com 3,5,8 e 10 objetivos. Os resultados mostraram características interessantes e resultados promissores em problemas com fronteiras Pareto irregulares. Como por exemplo nos problemas DTLZ5, IDTLZ1, MaF1, GPD1 e GPD2.

Palavras-chave: Otimização multi-objetivo, Decomposição, Algoritmo Evolucionário, Adaptação dos vetores de peso, Função de teste.



# Abstract

Multiobjective evolutionary algorithms (MOEA) present an interesting approach to solving different types of problems, known as multiobjective problems (MOP). The subcategory of MOEA with decomposition-based methods have been growing rapidly and many studies have shown that the distribution of weight vectors plays an interesting factor to obtain a uniform set of solutions. However, an uniform distribution of weight vectors at the beginning of evolution not always result in an uniform set of solutions in the objective space, as the results are highly dependent on the Pareto front shape. Irregularly shaped Pareto fronts (disconnected, inverted, etc.) generally do not contains all parts of the initial set of weight vectors. One approach to overcome this problem is to adapt the weight vectors to approximate the shape of the Pareto boundary. Aiming to contribute to the field of study, an algorithm based on decomposition that progressively adapts its weight vectors during the evolution process using a archive of nondominated solutions is proposed. The proposed algorithm is called Multi-objective Evolutionary Algorithm based on Decomposition with Local-Neighborhood Adaptation (MOEA/D-LNA). Subsequently, the proposed algorithm is compared to other algorithms from the literature in three sets of test functions, DTLZ, WFG, MaF and the one resulting from this research *Generalized Position-Distance* (GPD), with different weight vector initialization procedures in 3,5,8 and 10 objectives. The results have shown interesting characteristics and promising results on irregular Pareto fronts. For example on the problems DTLZ5, IDTLZ1, MaF1, GPD1 e GPD2.

*Keywords: Multiobjective Optimization, Decomposition, Evolutionary Algorithm , Weight vector adaptation, Benchmark function.*

# Lista de Ilustrações

Figura 1 – Exemplo comparando o resultado de uma execução do MOEA/D no problema de fronteira invertida IDTLZ1 e a distribuição dos vetores de pesos iniciais. Muitos vetores (em preto) não contemplam a fronteira Pareto (em azul) e ocorre o acúmulo de soluções nas bordas da fronteira	23
Figura 2 – Relação entre os vetores de pesos (em preto) e os diferentes formatos de fronteiras Pareto (em azul) em 3 dimensões: (a) Fronteira Regular, (b) Fronteira Invertida, (c) Fronteira Desconectada, (d) Fronteira Degenerada (Fonte: Própria).	36
Figura 3 – Diferença do intervalo das soluções para cada objetivo para a função de benchmark DTLZ7 (Fonte: Própria).	46
Figura 4 – Exemplo da construção das vizinhanças locais. Na figura, os pontos azuis representam soluções no arquivo, as setas tracejadas são os vetores de peso e a fronteira Pareto é representada como uma linha sólida azul (Fonte: Própria).	50
Figura 5 – Exemplo de três diferentes cones gerados pelo método WVG em um espaço tridimensional.	53
Figura 6 – Média do IGD para diferentes combinações dos parâmetros $k$ e $\phi$ do MOEA/D-LNA nos MOPs em diferentes dimensões (Fonte: Própria).	61
Figura 7 – Média do IGD com diferentes valores de $\delta_c$ do MOEA/D-LNA em MOPs e diferentes dimensões (Fonte: Própria).	62
Figura 8 – Variação da métrica de melhoria (IM) no problema DTLZ7 em 3 dimensões (Fonte: Própria).	63
Figura 9 – Variação do IGD no problema DTLZ7 em 3 dimensões (Fonte: Própria).	64
Figura 10 – Comparação entre as distribuições iniciais e finais dos vetores de peso do MOEA/D-LNA em diferentes problemas no espaço tridimensional: (a) IDTLZ1, (b) DTLZ7, (c) DTLZ5, (d) GPD02 (Fonte: Própria).	64
Figura 11 – Resultados com o melhor IGD para os problemas da DTLZ com o método de inicialização de vetores de peso $k$ -layers (Fonte: Própria).	67
Figura 12 – Resultados com o melhor IGD para os problemas da WFG com o método de inicialização de vetores de peso $k$ -layers (Fonte: Própria).	70

Figura 13 – Resultados com o melhor IGD para os problemas da GPD com o método de inicialização de vetores de peso $k$ -layers (Fonte: Própria). . . .	72
Figura 14 – Resultados com o melhor IGD para os problemas da GPD com o método de inicialização de vetores de peso Riesz S-Energy (Fonte: Própria). . . .	73
Figura 15 – Exemplos das fronteiras Pareto no espaço tridimensional das funções de teste propostas: a) GPD01 em comparação com a forma de um simplex regular em vermelho, b) GPD02 (Fonte: Própria). . . . .	92

# Lista de Tabelas

Tabela 1	– Número de variáveis de decisão ( $d$ ) e características da fronteira Pareto para cada problema. . . . .	59
Tabela 2	– Número de vetores de pesos e tamanho da população ( $N$ ) para cada método de inicialização. . . . .	60
Tabela 3	– Número médio de atualizações ativadas para cada problema da família DTLZ no MOEA/D-LNA. . . . .	66
Tabela 4	– Número médio de atualizações ativadas para cada problema da família WFG no MOEA/D-LNA. . . . .	69
Tabela 5	– Número médio de atualizações ativadas para cada problema da GPD do MOEA/D-LNA. . . . .	71
Tabela 6	– Número médio de atualizações ativadas para cada problema da família MaF no MOEA/D-LNA. . . . .	74
Tabela 7	– Resumo das vitórias (+), perdas (-) e empates (=) dos resultados de IGD de cada algoritmo do grupo 1 em comparação ao MOEA/D-LNA utilizando o teste estatístico não-paramétrico da soma dos postos de Wilcoxon com um nível de significância de 5%. . . . .	77
Tabela 8	– Resumo das vitórias (+), perdas (-) e empates (=) dos resultados de IGD de cada algoritmo do grupo 2 em comparação ao MOEA/D-LNA utilizando o teste estatístico não-paramétrico da soma dos postos de Wilcoxon com um nível de significância de 5%. . . . .	77
Tabela 9	– Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família DTLZ com o método de inicialização $k$ -layers para grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (-) e empates (=) para cada algoritmo contra MOEA/D-LNA. . . . .	94
Tabela 10	– Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família DTLZ com o método de inicialização $k$ -layers para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (-) e empates (=) para cada algoritmo contra MOEA/D-LNA. . . . .	95

Tabela 11 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família DTLZ com o método de inicialização Riesz S-Energy para grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA.	96
Tabela 12 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família DTLZ com o método de inicialização Riesz S-Energy para grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA.	97
Tabela 13 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família WFG com o método de inicialização $k$ -layers para grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA. . . . .	98
Tabela 14 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família WFG com o método de inicialização $k$ -layers para grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA. . . . .	99
Tabela 15 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família WFG com o método de inicialização Riesz S-Energy para grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA.	100
Tabela 16 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família WFG com o método de inicialização Riesz S-Energy para grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA.	101
Tabela 17 – Resultados de IGD (média e desvio padrão) para os algoritmos em 8 instâncias dos problemas da família GPD com o método de inicialização $k$ -layers para o grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA. . . . .	102

Tabela 18 – Resultados de IGD (média e desvio padrão) para os algoritmos em 8 instâncias dos problemas da família GPD com o método de inicialização <i>k</i> -layers para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA. . . . .	102
Tabela 19 – Resultados de IGD (média e desvio padrão) para os algoritmos em 8 instâncias dos problemas da família GPD com o método de inicialização Riesz S-Energy para o grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA.	103
Tabela 20 – Resultados de IGD (média e desvio padrão) para os algoritmos em 8 instâncias dos problemas da família GPD com o método de inicialização Riesz S-Energy para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA.	103
Tabela 21 – Resultados de IGD (média e desvio padrão) para os algoritmos em 48 instâncias dos problemas da família MaF com o método de inicialização <i>k</i> -layers para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA. . . . .	104
Tabela 22 – Resultados de IGD (média e desvio padrão) para os algoritmos em 48 instâncias dos problemas da família MaF com o método de inicialização Riesz S-Energy para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (–) e empates (=) para cada algoritmo contra MOEA/D-LNA.	106

# List of Algorithms

1	MOEA/D	30
2	MOEA/D-LNA	43
3	Cálculo das Vizinhanças Locais	51
4	Gerando Novos Vetores de Peso	53
5	Construção do Conjunto Final de Vetores de Peso	54

# Lista de Abreviaturas e Siglas

AMOEAD	<i>Adaptive Multiobjective Evolutionary Algorithm based on Decomposition</i>
ANSGA-III	<i>Adaptive Non-dominated Sorting Genetic Algorithm III</i>
APD	<i>Angle Penalized Distance</i>
C-MOGA	<i>Cellular Multiobjective Genetic Algorithm</i>
DE	<i>Differential Evolution</i>
EA	<i>Evolutionary Algorithms</i>
EAG-MOEAD	<i>External Archive Guided - Multiobjective Evolutionary Algorithm based on Decomposition</i>
EMOSA	<i>Evolutionary Multi-objective Simulated Annealing</i>
GPD	<i>Generalized Position Distance</i>
HV	<i>Hypervolume</i>
HypE	<i>Hypervolume-Based Many-Objective Optimization</i>
IGD	<i>Inverse Generational Distance</i>
IM	<i>Improvement Metric</i>
MaOEA	<i>Many-Objective Evolutionary Algorithm</i>
MaOEA-2ADV	<i>Many-objective Evolutionary Algorithm with two Types of Adjustments</i>
MaOP	<i>Manyobjective Optimization Problem</i>
MOEA/D	<i>Multiobjective Evolutionary Algorithm based on Decomposition</i>
MOEA/D-AWA	<i>MOEA/D with Adaptive Weight Adjustment</i>
MOEA/DD	<i>MOEA based on Dominance and Decomposition</i>



MOEA/D-DE	<i>Multiobjective Evolutionary Algorithm based on Decomposition Differential Evolution</i>
MOEA/D-DRA	<i>MOEA based on Dominance and Decomposition</i>
MOEAD-GR	<i>MOEA/D with Global Replacement</i>
MOEA/D-GRA	<i>MOEA/D with Generalized Resources Allocation</i>
MOEA/D-LNA	<i>Multiobjective Evolutionary Algorithm based on Decomposition with Local Neighborhoodbased Adaptaion</i>
MOEA/D-M2M	<i>Multiobjective Evolutionary Algorithm based on Decomposition MOP to MOP</i>
MOEA/D-RW	<i>Multiobjective Evolutionary Algorithm based on Decomposition with Random Weights</i>
MOGLS	<i>Multiobjective Genetic Local Search</i>
MOP	<i>Multiobjective Optimization Problem</i>
NSGA-III	<i>Non-dominated Sorting Genetic Algorithm III</i>
NSGA-II	<i>Non-dominated Sorting Genetic Algorithm II</i>
PBI	<i>Penalty-based Boundary Intesection</i>
PBEA	<i>Preference-based Evolutionary Algorithm</i>
PICEA-w	<i>Preference-Inspired co-evolutionary algorithm using Weight Vectors</i>
PF	<i>Pareto Front</i>
RVEA	<i>Reference Vector guided Evolutionary Algorithm</i>
SBX	<i>Simulated Binary Crossover</i>
SMS-EMOA	<i>S-metric Selection Evolutionary Multiobjective Algorithm</i>
SPEA2	<i>Strength Pareto Evolutionary Algorithm 2</i>
TCH	<i>Função de Agregação de Tchebycheff</i>
WVG	<i>Weight Vector Generator</i>
WS	<i>Função de Agregação Weighted Sum</i>

# Lista de Símbolos

$p_c$	Probabilidade de cruzamento
$p_m$	probabilidade de mutação
$B(i)$	Vizinhança de vetores
$T$	Tamanho da vizinhança
$\delta$	Probabilidade de compartilhamento da vizinhança
$nr$	Número máximo de soluções substituídas na vizinhança
$\theta$	Parâmetro de penalidade da PBI
$T_{Max}$	Número de gerações/avaliações
$t_{on}$	Numero mínimo de gerações/avaliações em que é permitido atualizar os vetores
$t_{off}$	Numero máximo de gerações/avaliações em que é permitido atualizar os vetores
$L(i)$	Vizinhança Local
$c$	Fator de multiplicação de vetores gerados no cone WVG
$\zeta$	Limiar para ajuste parâmetro da normlização
$\delta_c$	Porcentagem de vetores máxima que podem ser alterados
$\delta_r$	Probabilidade da solução em $s_i \in S$ ser alocada para um vetor da mesma vizinhança que o original
$\xi$	Ângulo de abertura do cone WVG
$\phi$	Fator de multiplicação do ângulo de abertura dos cones WVG
$k$	Tamanho da Vizinhança Local
$W$	Conjunto de vetores

$W'$  Conjunto de vetores após a etapa de remoção

$W_g$  Conjunto de novos vetores gerados

# Sumário

<b>1</b>	<b>Introdução</b>	<b>21</b>
1.1	Identificação do Problema e Motivação	22
1.2	Objetivos	24
1.3	Contribuições	24
1.4	Estrutura do Trabalho	25
<b>2</b>	<b>Fundamentação</b>	<b>26</b>
2.1	Conceitos Básicos	26
2.2	Multi-Objective Evolutionary Algorithms - MOEA	27
2.3	Multi-Objective Evolutionary Algorithm based on Decomposition - MOEA/D	28
2.3.1	Análise das Limitações do Framework MOEA/D	31
2.4	Estratégias de Alocação de Recursos	35
2.5	Plataforma PlatEMO	39
2.6	Considerações Parciais	40
<b>3</b>	<b>Algoritmo MOEA/D-LNA</b>	<b>41</b>
3.1	Introdução	41
3.2	Inicialização dos Vetores de Pesos	42
3.3	Normalização Dinâmica dos Objetivos	43
3.4	Frequência de Atualização dos Vetores de Peso	46
3.5	Manutenção do Arquivo de Soluções não Dominadas	48
3.6	Estratégia de Atualização dos Vetores de Pesos	48
3.6.1	Definição das Vizinhanças Locais	49
3.6.2	Remoção dos Vetores de Peso	51
3.6.3	Adição de Novos Vetores de Peso	51
3.6.4	Associação dos Vetores de Peso	54
3.7	Considerações Parciais	55
<b>4</b>	<b>Resultados</b>	<b>56</b>
4.1	Configuração Experimental	56
4.2	Configurações dos Problemas de Teste	58
4.3	Análise dos Parâmetros Adaptativos	59
4.4	Resultados e Discussões	65
4.4.1	Resultados nos Problemas DTLZ	65
4.4.2	Resultados nos Problemas WFG	66
4.4.3	Resultados nos Problemas GPD	69
4.4.4	Resultados nos Problemas MaF	71
4.4.5	Discussões dos Resultados	75
4.5	Considerações Parciais	78

<b>5</b>	<b>Conclusões</b>	<b>79</b>
5.1	Principais Resultados	80
5.2	Futuras Investigações	81
5.3	Publicações	81
	<b>Referências</b>	<b>83</b>
	<b>Apêndice A Problemas de Teste Propostos</b>	<b>91</b>
	<b>Apêndice B Tabelas de Resultados Completos de IGD</b>	<b>93</b>

# Capítulo 1

## Introdução

Diversos projetos, principalmente os de engenharia, são propostos visando solucionar problemas da melhor maneira possível. Estes problemas podem ser solucionados de várias formas distintas e geralmente envolvem a busca por certos parâmetros que entreguem o melhor custo-benefício, sendo necessário portanto um processo de otimização. A otimização pode ser definida como sendo o processo de obtenção dos melhores valores para os parâmetros de um determinado número de funções, conhecidas também como objetivos, que satisfaçam critérios do projeto de interesse [Rao, 2009]. Exemplos de problemas que podem ser caracterizados dessa maneira são controle de estoque, minimização de rotas e trajetórias de entregas, criação de novos produtos, planejamento de construções e muitos outros. Muitos destes projetos e pesquisas envolvem a otimização de mais de um objetivo, sendo conhecidos como *Multi-Objective Optimization Problems* (MOP).

Nos MOP, os objetivos possuem características conflitantes fazendo com que não haja uma única solução ótima, mas sim um conjunto de soluções que representam uma relação de compromisso, na qual a melhora em um objetivo pode resultar na piora de outro objetivo [Li et al., 2015]. O conjunto de soluções encontradas em que não haja outro conjunto melhor é conhecido como soluções Pareto ótimas.

Um subconjunto dos MOP, o qual vem sendo muito estudado recentemente, é o dos problemas de otimização com muitos objetivos, denominado de *Many-Objective Optimization Problems* (MaOP), sendo atualmente atribuído a problemas com quatro ou mais objetivos [Hua et al., 2021, Garza-Fabre et al., 2009]. Esta subdivisão acontece devido aos desafios enfrentados pelos algoritmos em uma alta dimensão, como por exemplo, o maior custo computacional para avaliação das soluções, dificuldades para visualização dos resultados e a grande maioria de algoritmos que são vistos como estado da arte e recomendados para solucionar MOP, enfrentam certas dificuldades nestes casos, sendo necessárias outras técnicas [Deb et al., 2014]. Os MaOP representam bem também problemas reais de engenharia, como no problema de abastecimento de água apresentado em [Kasprzyk et al., 2012] com diversos objetivos.

Dentre as maneiras computacionais de solucionar estes tipos de problemas, os algoritmos evolucionários, *Evolutionary Algorithms* (EA), são métodos que vêm sendo cada vez mais estudados devido a características favoráveis como eficiência, aplicabilidade em um grande número de problemas e em muitos casos necessitam de poucos ajustes para entregar boas soluções. Neste tipo de método, a ideia central é inspirada nos conceitos da teoria da evolução proposta por Darwin, em que uma população em um ambiente com recursos limitados compete por estes recursos e os mais adaptados sobrevivem e se reproduzem. Assim, a população passa a ser considerada como sendo o conjunto de soluções que se deseja obter, sendo avaliadas nas funções matemáticas que definem o modelo do problema. No contexto que envolve a solução de MOP, os algoritmos evolucionários são conhecidos como *Multi-Objective Evolutionary Algorithms* (MOEA).

## 1.1 Identificação do Problema e Motivação

Nos últimos anos na área de otimização, muitos trabalhos foram propostos buscando aperfeiçoar os algoritmos genéticos para que pudessem encontrar um conjunto de soluções que apresentassem boa convergência e distribuição em relação à fronteira Pareto do problema analisado. Os métodos baseados em dominância Pareto por exemplo, são muito reconhecidos pelo aspecto de eficiência, sendo aplicados para solucionar diversos tipos de MOP. Porém, é reconhecida a dificuldade enfrentada quando aplicada a MaOP, ocorrendo a deterioração dos resultados devido ao aumento da dimensionalidade do problema [Ishibuchi et al., 2008]. Dentre os diversos estudos em que são propostos métodos alternativos, aqueles baseados em funções de agregação vêm mostrando promissores resultados para a solução destes tipos de problemas.

Nos algoritmos baseados em decomposição/agregação, o problema multi-objetivo original é decomposto e resolvido em diversos sub-problemas mono-objetivo com o auxílio de um conjunto de vetores de pesos, que parametrizam as funções de agregação. Desta forma, os métodos baseados em função de agregação oferecem uma maneira eficiente de comparar soluções. Dentre os primeiros algoritmos baseados em funções de agregação, tem-se o MOGLS [Ishibuchi and Murata, 1998] e C-MOGA [Alba et al., 2007], porém a popularização deste tipo de método para solucionar MOP ocorreu somente a partir da introdução do *Multiobjective Evolutionary Algorithm based on Decomposition* (MOEA/D) [Zhang and Li, 2007].

O MOEA/D [Zhang and Li, 2007] utiliza um conjunto de vetores de pesos uniformemente distribuídos que auxiliam na aproximação de um conjunto final de soluções também uniformemente distribuídas. Contudo, este resultado dependerá fortemente do formato da fronteira Pareto que o problema apresenta. Em problemas de fronteira Pareto regular, ou seja, que são similares ou relativamente próximas do conjunto inicial de veto-

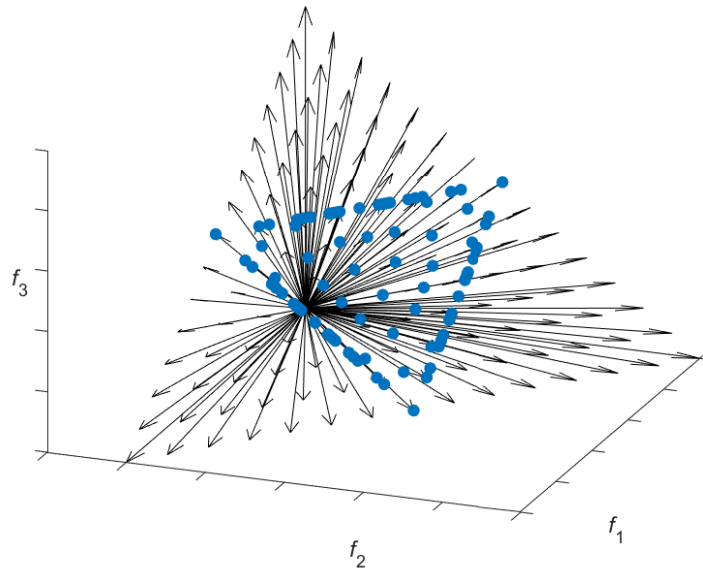


Figura 1 – Exemplo comparando o resultado de uma execução do MOEA/D no problema de fronteira invertida IDTLZ1 e a distribuição dos vetores de pesos iniciais. Muitos vetores (em preto) não contemplam a fronteira Pareto (em azul) e ocorre o acúmulo de soluções nas bordas da fronteira

res de peso, os resultados obtidos são satisfatórios e uniformemente distribuídos, já em fronteiras irregulares, onde a fronteira pode ser desconectada, degenerada, invertida, etc [Ishibuchi et al., 2016, Hua et al., 2021], há uma perda significativa da qualidade das soluções encontradas. Assim, dentre os fatores que podem comprometer os resultados nestes casos mais gerais, no que diz respeito aos algoritmos baseados em decomposição, muitos vetores de peso podem não estar localizados na região da fronteira Pareto, o que reduz a diversidade das soluções, gera um acúmulo de soluções nos bordos do Pareto aproximado obtido e conseqüentemente reduz a eficiência do algoritmo. Um exemplo pode ser observado na Figura 1 em que o resultado de uma execução do MOEA/D em um problema de fronteira invertida é apresentado junto ao conjunto de vetores de pesos iniciais. Portanto, uma distribuição uniforme dos vetores de pesos não garante uma boa distribuição das soluções encontradas pelo algoritmo [Ishibuchi et al., 2017b].

Com o intuito de solucionar estas dificuldades e melhorar a qualidade dos algoritmos baseados em decomposição, muitos estudos estão sendo desenvolvidos buscando adaptar o conjunto de vetores de pesos [Ma et al., 2020, de Farias and Araújo, 2021, Luque et al., 2020]. Dentre os desafios para realização da adaptação dos vetores de peso, podem ser citados os seguintes pontos principais:

1. A dificuldade de identificar as regiões promissoras, diferenciando regiões de difícil aproximação das regiões em que não há de fato soluções Pareto-ótimas;
2. Após realizada a identificação das regiões promissoras, tem-se o desafio de como



gerar os novos vetores de pesos;

3. Identificar o momento e a periodicidade adequados para realizar a adaptação dos vetores de pesos.

Partindo deste pressuposto, este trabalho tem por objetivo implementar uma versão adaptativa do algoritmo baseado na arquitetura de decomposição do MOEA/D onde são combinados também outros mecanismos que visam auxiliar neste processo.

## 1.2 Objetivos

O objetivo geral deste trabalho é propor um algoritmo genético multi-objetivo baseado em agregação com adaptação do conjunto de vetores de pesos, sendo capaz de lidar com problemas com fronteiras Pareto irregulares e MaOP. Para esta finalidade são definidos os seguintes objetivos específicos:

- Levantar as características e as limitações da arquitetura do MOEA/D.
- Levantar e comparar os métodos de inicialização de vetores de pesos, verificando sua influência nos algoritmos.
- Levantar algoritmos que implementam algum tipo de adaptação dos vetores de pesos.
- Propor um algoritmo baseado no MOEA/D que seja capaz de adaptar os vetores de pesos.
- Avaliar o modelo desenvolvido comparando o seu desempenho com outros MOEA estado-da-arte e com os que utilizam a arquitetura do MOEA/D com atualização dos vetores de pesos em conjuntos de problemas que apresentem diversas características.

## 1.3 Contribuições

A principal contribuição deste trabalho reside na proposição de um algoritmo baseado no MOEA/D que tem como principais estratégias para atualização dos vetores de peso (i) a utilização de soluções não-dominadas para auxiliar na identificação das melhores regiões da fronteira Pareto nas proximidades dos vetores de peso promissores e (ii) a metodologia de adição e remoção para atualizar o conjunto de vetores de peso. Cada um destes vetores é associado a uma vizinhança local contendo no máximo  $k$  soluções mais próximas. Assim, é utilizado o método de geração de vetores de pesos em um cone para criar novos vetores de pesos nas vizinhanças locais dos vetores promissores. O algoritmo é denominado de *MOEA/D with Local Neighborhood-based Adaptation* (MOEA/D-LNA).

Além disso, no MOEA/D-LNA são incorporadas e adaptadas outras metodologias como a normalização dinâmica do espaço de objetivos [He et al., 2020], o ajuste dinâmico de parâmetro de normalização, métrica de medição de estagnação da população com o intuito de auxiliar o controle da frequência de atualização dos vetores de pesos [Camacho et al., 2019] e técnica de manutenção do conjunto de soluções não-dominadas [Li et al., 2016].

São propostas também novas funções de *benchmark* que utilizam como base o gerador de funções de testes conhecido como *Generalized Position Distance* (GPD) [Menechini et al., 2020a]. Estas funções foram desenvolvidas com o intuito de explorar os desafios encontrados nos problemas de fronteira Pareto irregular e foram denominadas de GPD.

## 1.4 Estrutura do Trabalho

Este trabalho está organizado em cinco capítulos, incluindo esta introdução que tem por objetivo apresentar o problema e as ideias centrais do trabalho.

No Capítulo 2 é apresentada a fundamentação teórica que serviu como base para desenvolver o trabalho. Neste são apresentados os conceitos de otimização, a arquitetura do MOEA/D, assim como identificados os seus pontos de melhoria. Também são apresentados os principais MOEA encontrados na literatura que aplicam técnicas de adaptação de vetores de peso.

No Capítulo 3 é apresentado o desenvolvimento do algoritmo proposto. Neste, são especificados os componentes auxiliares e detalhadas as etapas para a atualização dos vetores de pesos.

No Capítulo 4, o algoritmo proposto no Capítulo 3 é comparado com outros MOEA extraídos da literatura em alguns conjuntos de funções de *benchmark*, conhecidos como DTLZ [Deb et al., 2005, Jain and Deb, 2014], WFG [Huband et al., 2006] e MaF [Cheng et al., 2017], além das funções propostas GPD (Apêndice A), com 3, 5, 8 e 10 objetivos. Os resultados são apresentados sob a perspectiva de análises qualitativas e quantitativas utilizando a métrica conhecida como *Inverse Generational Distance* (IGD) [Veldhuizen and Lamont, 1998] e testes estatísticos.

Finalmente, no Capítulo 5 são apresentadas as conclusões acerca do trabalho desenvolvido, contrapondo os objetivos iniciais, assim como apresentadas propostas para continuidade da pesquisa em futuros trabalhos.

# Capítulo 2

## Fundamentação Teórica

Esta seção tem por objetivo reunir os conceitos teóricos que foram utilizados para embasar os estudos apresentados neste trabalho. Na Seção 2.1 tem-se os conceitos e termos básicos na área de otimização. A Seção 2.2 contextualiza as principais áreas dos algoritmos multi-objetivo. Na sequência, na Seção 2.3 é feita uma revisão da arquitetura do algoritmo MOEA/D, que é utilizado como base para o desenvolvimento do MOEA/D-LNA, e os principais pontos de melhoria, assim como as pesquisas relacionadas. Finalmente, na Seção 2.4 é apresentado o problema principal da adaptação dos vetores de peso e as principais abordagens encontradas na literatura.

### 2.1 Conceitos Básicos

Um problema de otimização multi-objetivo pode ser representado na seguinte notação matemática:

$$\begin{aligned} \min / \max \mathbf{F}(\mathbf{x}) &= (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x}))^T \\ \text{Sujeito a } \mathbf{x} &\in \Omega \end{aligned} \tag{2.1}$$

Com  $\Omega$  sendo o espaço de busca e  $\mathbf{x} = (x_1, x_2, \dots, x_d)^T$  o vetor das variáveis de decisão com  $d$  variáveis no espaço  $\mathbb{R}^d$  e  $\mathbf{F} : \Omega \rightarrow \mathbb{R}^M$ , com  $M$  objetivos em  $\mathbb{R}^M$ . Também há a possibilidade de estar sujeito a restrições de igualdade ou desigualdade. No caso dos problemas com dimensões superiores, *Many-Objective* (MaOP),  $M > 3$ .

Duas soluções candidatas podem ser avaliadas através do conceito de dominância Pareto. Portanto, para duas variáveis  $\mathbf{u} = (u_1, u_2, \dots, u_d)^T$  e  $\mathbf{v} = (v_1, v_2, \dots, v_d)^T$ , tem-se que  $\mathbf{u}$  domina  $\mathbf{v}$  se e somente se  $\mathbf{u}$  é melhor ou igual a  $\mathbf{v}$  em todos os objetivos do MOP e tem-se ao menos um dos objetivos em que  $\mathbf{u}$  seja melhor que  $\mathbf{v}$ , (representado por  $f(\mathbf{u}) \preceq f(\mathbf{v})$ ). Em um problema de minimização, pode ser escrita matematicamente a dominância Pareto

como:

$$\begin{aligned} f_i(\mathbf{u}) &\leq f_i(\mathbf{v}), \quad \forall i \in \{1, 2, M\} \\ f_j(\mathbf{u}) &< f_j(\mathbf{v}), \quad \text{para algum } j \in \{1, 2, M\} \end{aligned} \quad (2.2)$$

As soluções que não são dominadas por nenhuma outra solução são atribuídas ao conjunto Pareto-ótimo, ou seja, são soluções Pareto-ótimas. Define-se matematicamente o conjunto de soluções Pareto-ótimo como sendo:

$$\mathcal{P} = \{\mathbf{x} \in \Omega \mid \nexists \mathbf{y} \in \Omega, \mathbf{F}(\mathbf{y}) \preceq \mathbf{F}(\mathbf{x})\} \quad (2.3)$$

A fronteira Pareto é definida como sendo o mapeamento do conjunto Pareto-ótimo no espaço de objetivos:

$$\mathcal{FP} = \mathbf{F}(\mathcal{P}) = \{\mathbf{F}(\mathbf{x}) \mid \mathbf{x} \in \mathcal{P}\} \quad (2.4)$$

## 2.2 Multi-Objective Evolutionary Algorithms - MOEA

Os algoritmos evolucionários, são métodos computacionais baseados na evolução dos seres vivos e possuem a finalidade de resolver problemas em diversas áreas. São algoritmos baseados em populações que simulam os processos de nascimento, seleção, reprodução e adaptação dos seres. Ao longo dos anos foram se tornando cada vez mais populares, principalmente pela capacidade de aproximar conjuntos de soluções Pareto-ótimas em apenas uma execução e pela grande capacidade de solucionar os MOP, onde são conhecidos como *Multi-Objective Evolutionary Algorithms* (MOEA) [Coello, 1999, Zhou et al., 2011].

Os MOEA podem ser inicialmente classificados em quatro categorias principais que definem o tipo de abordagem adotada, sendo estas: as baseadas em dominância Pareto, em indicadores, em preferência e em métodos que utilizam funções de agregação [Li et al., 2015, Ishibuchi et al., 2008]. Os métodos baseados em dominância Pareto, de forma simplificada, utilizam o conceito de frentes de dominância para selecionar as soluções não dominadas e ordenando as melhores soluções, como nos populares algoritmos: *Nondominated Sorting Genetic Algorithm* (NSGA-II) [Deb et al., 2002] e o *Improved Strength Pareto Evolutionary Algorithm* (SPEA2) [Zitzler et al., 2001]. Já os métodos baseados em indicadores, como o próprio nome ressalta, utilizam indicadores de qualidade, como por exemplo o Hipervolume e o IGD, para avaliar a convergência das soluções encontradas e evoluir a população na direção correta, a exemplo destes algoritmos tem-se o *S-metric Selection Evolutionary Multiobjective Algorithm* (SMS-EMOA) [Emmerich et al., 2005] e o *Hypervolume-Based Many-Objective Optimization* (HypE) [Bader and Zitzler, 2011]. Por sua vez, os métodos baseados em preferência são algoritmos em que se tem uma ideia do conjunto de soluções desejadas, geralmente a priori, e estas são usadas como pontos

de referência para que a população evolua, como por exemplo o *Preference-based Evolutionary Algorithm* (PBEA) [Thiele et al., 2009]. Finalmente, os métodos baseados em agregação são algoritmos que utilizam funções de agregação para decompor o problema multi-objetivo em vários subproblemas. Um dos exemplos mais conhecidos é o *Multiobjective Evolutionary Algorithm Based on Decomposition* (MOEA/D) [Zhang and Li, 2007]. Há também pesquisas em que são mescladas duas ou mais destas abordagens, tendo como exemplo o algoritmo baseado em decomposição que incorpora uma região de interesse no processo de evolução [Meneghini et al., 2020b] e o *Multiobjective Evolutionary Algorithm Based on Dominance and Decomposition* (MOEA/DD) [Li et al., 2015] que utiliza frentes de dominância em conjunto com os conceitos de decomposição.

Em se tratando dos MaOP, muitos algoritmos clássicos e reconhecidos pela ótima eficiência em dois ou três objetivos se mostram ineficazes [Ishibuchi et al., 2011]. Com o aumento da dimensão, os algoritmos baseados em dominância, por exemplo, sofrem uma degradação nos resultados, pois quase todas as soluções da população se tornam não dominadas, reduzindo assim a pressão seletiva [Ishibuchi et al., 2008]. Nos algoritmos baseados em indicadores, apesar de não apresentarem os problemas de perda de pressão seletiva, eles ainda são afetados pelo problema da dimensionalidade, e neste caso, o custo computacional cresce de forma exponencial com o aumento do número de objetivos [Li et al., 2015]. Já os algoritmos baseados em preferência necessitam de alguma informação a priori sobre a região de interesse, o que muitas das vezes é desconhecida [Trivedi et al., 2016]. Já os métodos baseados em funções de decomposição/agregação vêm se tornando muito populares para resolução de MOP e também apresentando promissores resultados no contexto dos MaOP [Ishibuchi et al., 2015].

## 2.3 Multi-Objective Evolutionary Algorithm based on Decomposition - MOEA/D

O MOEA/D [Zhang and Li, 2007] é o algoritmo evolucionário que popularizou a utilização de funções de agregação para resolução de MOP. Neste algoritmo, o MOP é decomposto em  $N$  sub-problemas mono-objetivo utilizando algum método de decomposição. Cada um desses subproblemas é associado a uma única função de agregação e a um vetor de pesos diferente. Para as funções de agregação, os autores propuseram a utilização de três funções de agregação distintas: a *Weighted Sum* (WS), a *Tchebycheff* (TCH) e a *Penalty-Based Boundary Intersection* (PBI).

Para o método de agregação utilizando a função *Weighted Sum* e dado um conjunto de vetores  $W$ , tem-se a solução ótima para o seguinte problema de otimização escalar

através da Equação (2.5):

$$g(\mathbf{x}|\mathbf{w}) = \sum_{i=1}^M w_i f_i(\mathbf{x}) \quad (2.5)$$

Sujeito a  $\mathbf{x} \in \Omega$

Esta função de escalarização apresenta resultados satisfatórios em um problema de minimização quando a fronteira Pareto é convexa. Porém, já não é recomendada para os casos em que a fronteira é não-convexa, pois nem toda solução Pareto-ótima pode ser encontrada [Jiang et al., 2018].

Na abordagem com a função de escalarização de *Tchebycheff*, o problema é representado através da Equação (2.6):

$$\min g(\mathbf{x}|\mathbf{w}, \mathbf{z}^*) = \max\{w_i | f_i(\mathbf{x}) - z_i^* |\} \quad (2.6)$$

Sujeito a  $\mathbf{x} \in \Omega$

em que  $\mathbf{z}^* = (z_1^*, \dots, z_M^*)^T$  é o ponto estimado ideal do MOP. Neste caso, é definido que para cada ponto ótimo  $\mathbf{x}^*$  existirá um vetor de pesos  $\mathbf{w}$  tal que este ponto  $\mathbf{x}^*$  é uma solução ótima da Equação (2.6) e conseqüentemente, é uma solução Pareto ótima. Esta função de escalarização tem como vantagem na abordagem de problemas com fronteiras não-convexas quando comparada com a WS.

Finalmente, utilizando a abordagem com a função *Penalty-Based Boundary Intersection* (PBI), cujas equações são apresentadas na Equação (2.7):

$$\begin{aligned} \min g(\mathbf{x}|\mathbf{w}, \mathbf{z}^*) &= d_1 + \theta d_2 \\ \text{Sujeito a } \mathbf{x} &\in \Omega \\ d_1 &= \frac{\|(\mathbf{z}^* - \mathbf{F}(\mathbf{x}))^T \mathbf{w}\|}{\|\mathbf{w}\|} \\ d_2 &= \left\| \mathbf{F}(\mathbf{x}) - \left( \mathbf{z}^* - d_1 \frac{\mathbf{w}}{\|\mathbf{w}\|} \right) \right\| \end{aligned} \quad (2.7)$$

A PBI envolve a utilização de duas métricas calculadas no espaço de objetivos que medem a distância entre a solução e o ponto ideal e a distância da projeção da solução  $\mathbf{x}$  à linha formada entre o ponto ideal  $\mathbf{z}^*$  e o vetor de referência  $\mathbf{w}$ , denominadas de  $d_1$  e  $d_2$  respectivamente. O balanceamento entre essas duas medidas é realizado através do parâmetro de penalidade  $\theta$ . Com vantagens em problemas de fronteira Pareto convexa, em que maiores valores de  $\theta$  auxiliam no aumento da diversidade, enquanto menores valores contribuem para convergência [Jiang et al., 2018].

Na seqüência, no Algoritmo 1 é apresentada a estrutura do método MOEA/D. Durante a etapa inicial do algoritmo, são inicializados de forma aleatória a população de indivíduos ( $P$ ) e o conjunto de vetores de peso (linhas 1-2). No trabalho original, os

**Algorithm 1** MOEA/D**Input:**  $N, M, T$ 


---

```

1: Inicializar a População  $\mathbf{P}$ ;
2: Inicializar o conjunto de vetores de pesos  $\mathbf{W}$ ;
3: Calcular as vizinhanças  $\mathbf{B}(i)$  dos vetores de pesos  $\mathbf{w}_i \in \mathbf{W}$ ;
4: Inicializar o conjunto  $\mathbf{EP}$  com as soluções não-dominadas de  $\mathbf{P}$ ;
5: while critério de parada não atendido do
6:   for cada sub-problema  $i = 1 : N$  do
7:      $\mathbf{p} \leftarrow$  gera uma nova solução de  $\mathbf{B}(i)$ ;
8:     if  $g(\mathbf{p}|\mathbf{w}_i, \mathbf{z}^*) \leq g(\mathbf{x}_i|\mathbf{w}_i, \mathbf{z}^*)$  then
9:        $\mathbf{x}_i \leftarrow \mathbf{p}$ ;
10:    end if
11:    Atualizar o ponto ideal  $\mathbf{z}^*$ ;
12:    for cada  $j \in \mathbf{B}(i)$  do
13:      if  $g(\mathbf{p}|\mathbf{w}_j, \mathbf{z}^*) \leq g(\mathbf{x}_j|\mathbf{w}_j, \mathbf{z}^*)$  then
14:         $\mathbf{x}_j \leftarrow \mathbf{p}$ ;
15:      end if
16:    end for
17:  end for
18:  Atualizar as piores soluções na  $\mathbf{EP}$ 
19: end while
20: return  $\mathbf{EP}$ 

```

---

autores adotaram o método conhecido como simplex-lattice [Das and Dennis, 2000] para a geração uniforme dos vetores de peso. Neste, os vetores de peso  $\mathbf{w} = (w_1, w_2, \dots, w_M)^T$  satisfazem as seguintes relações matemáticas:

$$\sum_{i=1}^M w_i = 1, \text{ com } i = 1, \dots, M \quad (2.8)$$

$$w_i \in \left\{ 0, \frac{1}{H}, \frac{2}{H}, \dots, \frac{H}{H} \right\}, \text{ } i = 1, 2, \dots, M \quad (2.9)$$

em que  $H$  é uma variável inteira definida pelo usuário. O número total de vetores de peso gerados através deste método depende do parâmetro  $H$  e da dimensão do problema,  $M$  e pode ser calculado através de  $N = \binom{H+M-1}{M-1}$ . Ainda na inicialização, é calculada a vizinhança dos vetores de peso (linha 3). Definida como  $\mathbf{B}(i) = \{i_1, \dots, i_T\}$ , a vizinhança de cada sub-problema é determinada através dos  $T$  vetores de peso mais próximos utilizando a distância Euclidiana entre todos os vetores de peso do conjunto  $\mathbf{W}$ . Também, a população externa ( $\mathbf{EP}$ ) é inicializada contendo as soluções iniciais não-dominadas (linha 4).

A partir de todos os requisitos inicializados, segue o processo de atualização enquanto o critério de parada não for atendido. Para cada um dos  $N$  sub-problemas, são primeiramente escolhidos de forma aleatória dois indivíduos da vizinhança  $\mathbf{B}(i)$  para cruzarem e gerarem um descendente,  $\mathbf{p}$ , sofrendo também uma mutação (linha 7). No trabalho original são utilizados o *Simulated Binary Crossover* (SBX) e o *Polynomial Mutation* como operadores genéticos. Na sequência, a aptidão do descendente é calculada,

substituída no sub-problema atual, caso melhor e é verificada se esta nova solução é melhor que o ponto ideal estimado, caso sim, será também atualizado (linhas 8-11). Após, para cada individuo da vizinhança atual, é verificado se  $\mathbf{p}$  é melhor, através do valor da função de escalarização (linhas 12-17). Por fim, após todos os  $N$  sub-problemas terem sido verificados, atualiza-se a população externa com as soluções não-dominadas (linha 18).

### 2.3.1 Análise das Limitações do Framework MOEA/D

Apesar da estrutura do MOEA/D ter se tornado popular nos últimos anos e se mostrar promissora na solução de MaOP, mesmo não tendo sido projetada para tal, ela apresenta limitações e pontos que ainda podem ser melhorados [Ishibuchi et al., 2015]. Deste modo, muitos outros trabalhos foram conduzidos visando corrigir os pontos de falhas, superar as limitações e conseqüentemente, melhorar a qualidade dos resultados em diversos tipos de problemas, gerando assim, novas versões do próprio MOEA/D e de algoritmos baseados em decomposição de maneira geral.

Da literatura, é possível identificar e agrupar os principais pontos de melhoria da estrutura do MOEA/D nas seguintes categorias: 1) inicialização dos vetores de peso, 2) funções de decomposição, 3) alternativas de métodos de seleção, 4) substituição e variação de operadores genéticos e 5) estratégias de alocação de recursos, sendo este último explicado na próxima seção [Trivedi et al., 2016].

1. **Inicialização de vetores de peso:** No trabalho original, o MOEA/D [Zhang and Li, 2007] utiliza o método conhecido como *simplex-lattice design* [Das and Dennis, 2000] para gerar os vetores de pesos, como apresentado anteriormente. Mesmo sendo muito utilizada na literatura em MOEA baseados em decomposição e apresentar uma distribuição uniforme de vetores pesos, o número total de vetores de peso e conseqüentemente o tamanho da população é altamente dependente da dimensão do problema. Assim, estes parâmetros não podem ser facilmente controlados, mesmo variando o parâmetro  $H$  do método de inicialização. Analisando alguns exemplos de configurações para a inicialização dos vetores de pesos, pode-se entender estas dificuldades. No caso em que  $H < M$  os vetores são gerados apenas na borda do simplex, deixando seu interior vazio. Para os casos em que  $H \geq M$ , obtém-se um grande número de vetores e de população  $N$  em dimensões maiores. Por exemplo, para um problema de 8 objetivos e configurando o parâmetro  $H = 8$ , tem-se  $N = 6.435$ , ou seja, aumentando de forma não linear com o número de objetivos [Ishibuchi et al., 2016]. Visando estes aspectos, vários trabalhos na literatura propõem novas formas de gerar os vetores de pesos. Alguns exemplos são apresentados a seguir:

- a) No algoritmo MOEA/DD, proposto em Deb et al. [2014] em que é mesclado



os conceitos de decomposição e dominância, é apresentado um novo método de geração de vetores de pesos, focado em problemas multi-objetivo. Para isso, os autores apresentam o método denominado de *Two-layer*, onde são gerados dois conjuntos de vetores de pesos, ditos de camada externa e interna, utilizando o método do simplex-lattice [Das and Dennis, 2000] com parâmetros  $H$  distintos. Em seguida, utilizam uma transformação de coordenadas para reduzir o conjunto da camada interna e o combina com o do conjunto da camada externa. O conjunto completo terá um total de vetores gerados em cada um dos conjuntos, ou seja,  $N1 + N2 = N$  vetores de pesos. Com este método é apresentado um conjunto de vetores relativamente mais dispersos e com mais flexibilidade para maiores dimensões, melhorando os resultados.

- b) No trabalho de Jiang and Yang [2017] o conceito de camada apresentado no item anterior é expandido e é proposto o método de geração de vetores de pesos em  $k$  camadas, denominado de *k-Layers*. Para construir o conjunto de vetores de pesos, é utilizado um ponto central,  $\mathbf{C} = (1/M, 1/M, \dots, 1/M)$  e as  $i$ -ésimas referências extremas, que interceptam os eixo coordenados,  $\mathbf{B}_i = (b_1, b_2, \dots, b_M)$  onde  $b_i = 1$  e  $b_j = 0$  para todo  $j \neq i, 1 \leq j \leq M, 1 \leq i \leq M$ . Na sequência, o simplex é dividido em  $M$  sub-simplexes com os pontos  $\mathbf{C}, \mathbf{B}^i$  e  $\mathbf{B}^{i+1}$  para todo  $1 \leq i \leq M - 1$ . Através da Equação 2.10:

$$\mathbf{D}_i^r = \mathbf{C} + \frac{r}{k}(\mathbf{B}^i - \mathbf{C}), \quad r \in 1, \dots, k \quad (2.10)$$

são gerados  $k$  pontos entre o ponto central,  $\mathbf{C}$  e as  $i$ -ésimas referências extremas, denominados de  $\mathbf{D}_i^r$ . Finalmente, são gerados os pontos internos para cada  $r$ -ésima camada de cada sub-simplex. O  $t$ -ésimo ponto,  $\hat{\mathbf{D}}_{i,r}^t$  é dado através de:

$$\hat{\mathbf{D}}_{i,r}^t = \mathbf{D}_i^r + \frac{t}{t+1}(\mathbf{D}_{i+1}^r - \mathbf{D}_i^r), \quad t \in 1, \dots, k \quad (2.11)$$

O *k-layers* apresenta uma boa distribuição de vetores de pesos e escala ainda melhor com a dimensão, sendo mais fácil gerar um número menor de vetores.

- c) Para gerar um conjunto de vetores de pesos bem distribuídos no espaço, os autores em Blank et al. [2020] propõem um novo método para a geração dos vetores de pesos que apresenta um conjunto arbitrário de pontos bem distribuídos no simplex. Neste método é utilizado o conceito de energia potencial em um espaço multidimensional, denominado de *Riesz s-Energy*. O método *Riesz s-Energy* gera um conjunto de vetores de peso uniformemente distribuídos através do conceito de minimização da energia potencial total entre dois pontos no espaço. A ideia é que a energia potencial relacionada a dois elementos seja proporcional ao inverso da distância entre eles e, ao minimizá-la, o

procedimento geraria um conjunto de pontos uniformemente distribuídos no espaço. A função adaptada para calcular a energia potencial é dada por:

$$U_T(\mathbf{z}) = \frac{1}{2} \sum_{i=1}^M \sum_{\substack{j=1 \\ j \neq i}}^M \frac{1}{\|z_i - z_j\|^s} \quad (2.12)$$

onde  $\mathbf{z}$  é uma matriz  $N \times M$  que minimiza  $U_T$  e tem todos os vetores  $\mathbf{z}_i$  contidos em um simplex unitário. Assim, ao final, o algoritmo produzirá um conjunto de vetores de peso, no caso a matriz  $\mathbf{z}$ , que é uniformemente distribuída no espaço  $M$ -dimensional. O método possibilita controle total do número de vetores de pesos em qualquer dimensão e apresenta resultados uniformemente distribuídos.

**2. Funções de agregação** A função de agregação é uma componente fundamental da estrutura que irá influenciar a capacidade de busca do algoritmo baseado em decomposição [Ishibuchi et al., 2010]. No estudo do MOEA/D [Zhang and Li, 2007], os autores testam três diferentes métodos de agregação: *Weighted Sum* (WS), *Tchebycheff* (TCH) e *Penalty-Based Boundary Intersection* (PBI). Porém, é verificado que o WS apresenta bons resultados para fronteiras convexas mas é incapaz de aproximar as não-convexas e o PBI, apesar de ser bastante utilizado em vários trabalhos, apresenta a dificuldade extra na hora de configurar o seu fator de penalidade,  $\theta$  [Wang and Zhang, 2015]. Outros estudos realizados apontam que os métodos que utilizam o ponto de referência estimado  $\mathbf{z}$ , como o TCH e PBI, apresentam baixa performance em MaOP devido à grande diferença entre o valor real do ponto utópico e seu valor estimado [Sato, 2014]. Na sequência são apresentados alguns trabalhos que propuseram métodos alternativos de decomposição:

- a) Em [Sato, 2014], é realizada uma alteração na função de agregação do MOEA/D baseado no PBI. Nesta versão é proposto o “PBI invertido”, onde é utilizado o ponto antiutópico para evoluir as soluções, através da maximização da função de agregação. Apresentando resultados promissores e melhorias quando aplicado a MaOPs.
- b) Alguns trabalhos propõem a combinação de funções de agregação. Em [Ishibuchi et al., 2010], é realizada a combinação das funções WS e TCH em dois casos: utilizando um sistema multi-grid, onde cada função possui seu conjunto de vetores de pesos e em um segundo caso em que os vetores de pesos são compartilhados, dito *single-grid*. O estudo aponta melhorias com relação ao MOEA/D original [Zhang and Li, 2007].
- c) O trabalho apresentado em [Cheng et al., 2016], utiliza um método de agregação que visa balancear a convergência e a diversidade, denominado de *Angle*

*Penalized Distance* (APD). O critério de convergência é calculado com relação a distância das possíveis soluções para  $\mathbf{z}$ , e já o de diversidade, é calculado com relação ao ângulo agudo entre a solução e os vetores de referência. Diferentemente do MOEA/D-PBI original [Zhang and Li, 2007], o parâmetro de penalidade não é fixo e se adapta ao processo de evolução, dependendo somente de um parâmetro determinado pelo usuário,  $\alpha$ . Os autores apresentam resultados satisfatórios para diferentes configurações de problemas e objetivos sem a necessidade de alterar o parâmetro  $\alpha$ .

3. **Métodos alternativos de seleção e substituição:** Na estrutura do MOEA/D proposto por Zhang and Li [2007] o conceito de vizinhança é importante para a convergência, porém uma única solução tem a possibilidade de substituir a vizinhança inteira. Conseqüentemente, a seleção das soluções pai ocorre através dos indivíduos contidos na vizinhança, o que pode deteriorar a diversidade nestes casos [Ishibuchi et al., 2016]. Neste aspecto, são apresentados abaixo algumas outras propostas:

- a) Em uma nova versão, conhecida como MOEA/D-DE [Li and Zhang, 2009], os autores também propõem alternativas para corrigir e melhorar alguns pontos do MOEA/D original [Zhang and Li, 2007]. São propostas duas alterações: Na primeira, dada uma probabilidade, os pais poderão ser selecionados ou da vizinhança  $\mathbf{B}(i)$  ou do conjunto global de soluções, permitindo aumentar a diversidade. Para o segundo caso, é implementada uma mecânica em que um parâmetro controla o número de soluções que serão atualizadas por uma nova solução caso esta tenha valor de função de decomposição melhor. Desta forma, pode-se garantir que somente um certo número de soluções poderão ser substituídas por cada geração, evitando perda da diversidade devido a um super indivíduo logo no início da otimização.
- b) No algoritmo denominado de MOEA/D-GR Wang et al. [2014], é proposta a utilização de duas vizinhanças no processo de seleção e substituição. Neste, há uma vizinhança com a finalidade de identificar os melhores indivíduos para cruzamento, e uma outra exclusiva de onde é realizada a substituição. Assim, uma nova solução é gerada a partir das soluções pais da sua vizinhança e ela poderá substituir, de forma mais eficiente, alguma outra solução pertencente a outra vizinhança.
- c) Os autores do algoritmo MOEA/D-TPN [Jiang and Yang, 2016], além de outras melhorias, propõem a utilização de um método de cruzamento “guiado por nichos”, onde são calculados o grau de pertinência de cada indivíduo à sua vizinhança. Este valor sendo maior que dado limite, indica que o indivíduo é similar aos da sua vizinhança, assim, são escolhidos os indivíduos de outra

vizinhança para realizar o cruzamento. Esta metodologia, busca melhorar a diversidade das soluções durante a evolução.

4. **Diferentes operadores genéticos:** Outro ponto interessante e abordado envolve o estudo de variações dos operadores genéticos, recombinação SBX e a mutação polinomial, implementados no MOEA/D. De forma geral, este é um fator importante nos MOEA e os operadores são recomendados de acordo com o tipo de problema em que se deseja utilizar [Agrawal et al., 2000]. Alguns estudos implementaram operadores variados no MOEA/D com o objetivo de melhorar a performance em MaOP:

- a) Além da adaptação do método de seleção, na variação MOEA/D-DE [Li and Zhang, 2009] é utilizado o operador genético conhecido como *differential evolution* (DE), o qual dá o nome da versão melhorada. O intuito dos autores foi melhorar a eficiência dos cruzamentos do MOEA/D original para poder lidar com PF complexos.
- b) Zhou et al. [2012] utilizaram a estrutura do MOEA/D em conjunto com um modelo de distribuição Gaussiana multivariada. Neste algoritmo, o modelo probabilístico é gerado a partir das soluções da vizinhança ou selecionadas aleatoriamente da população, assim, uma nova solução é amostrada deste modelo.
- c) O estudo de Li and Landa-Silva [2011] propõe a combinação da estrutura do MOEA/D [Zhang and Li, 2007] com o algoritmo conhecido como *Simulated Annealing*, denominado de EMOSA. O EMOSA, não possui um operador de cruzamento, ao invés disso, o *simulated annealing* é utilizado para uma busca local com a finalidade de melhorar a solução atual e então substituir os indivíduos da vizinhança  $\mathbf{B}(i)$ .

## 2.4 Estratégias de Alocação de Recursos

Conforme apresentado no início deste capítulo, o MOEA/D [Zhang and Li, 2007] leva em consideração que todos os subproblemas são iguais e aloca a mesma quantidade de recursos para cada um deles, desta forma sendo projetado para encontrar um conjunto de soluções que é próximo aos vetores gerados durante o processo de inicialização. Portanto, uma boa distribuição do conjunto de vetores de pesos no espaço de objetivos deveria resultar em uma distribuição uniforme das soluções na fronteira Pareto aproximada. Para problemas com fronteira Pareto Regular essa condição geralmente é alcançada, porém este não é o caso quando aplicado a problemas de fronteira Pareto complicada (degenerada, desconexa, invertida, etc) [Ishibuchi et al., 2017b, Wang et al., 2015, Li et al., 2016]. Nestes últimos casos, haverá regiões do conjunto inicial de vetores de pesos que não

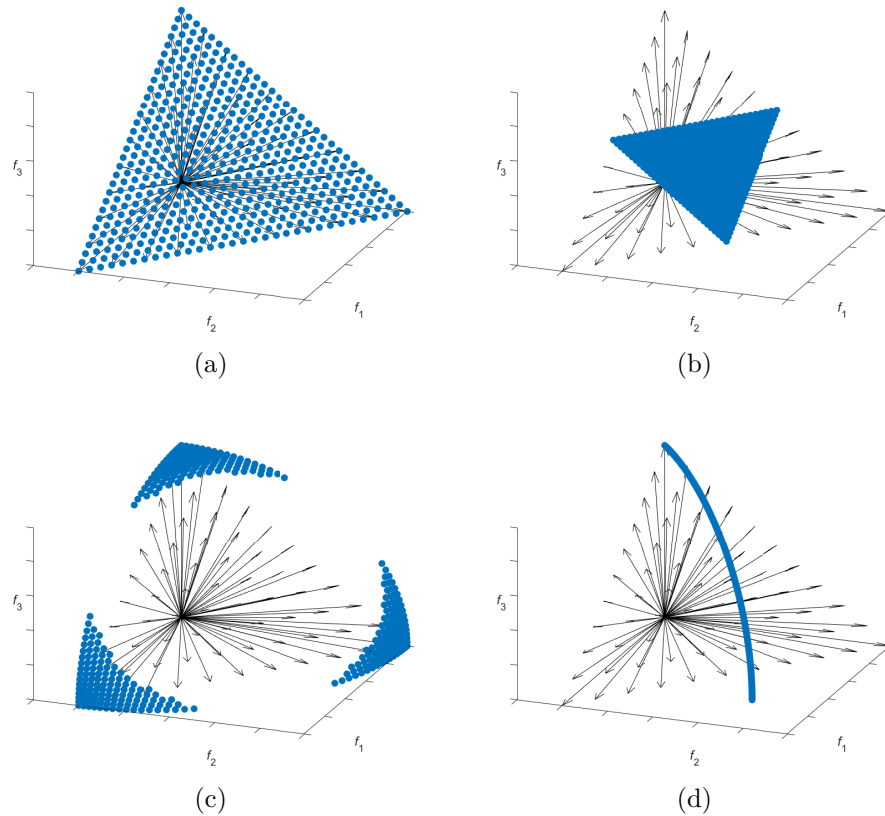


Figura 2 – Relação entre os vetores de pesos (em preto) e os diferentes formatos de fronteiras Pareto (em azul) em 3 dimensões: (a) Fronteira Regular, (b) Fronteira Invertida, (c) Fronteira Desconectada, (d) Fronteira Degenerada (Fonte: Própria).

contemplarão nenhuma solução Pareto-ótima [Hua et al., 2019], sendo mais difíceis de serem aproximadas quando os recursos são fixos em cada subproblema [Zhou and Zhang, 2015]. A exemplo desta característica, a Figura 2 demonstra um conjunto de vetores uniformemente distribuídos e problemas com diferentes tipos de fronteira Pareto em 3 dimensões. Fica possível notar, que nas fronteiras regulares todos os vetores de pesos podem levar a uma solução Pareto-ótima, já para as irregulares, é possível perceber que existem vetores que não contemplam nenhuma parte da fronteira.

Uma promissora vertente de estudos na área dos MOEA é a alocação de recursos relacionados a atualização de vetores, que não somente envolve os algoritmos baseados na arquitetura do MOEA/D [Zhang and Li, 2007] mas também outros tipos de algoritmos que combinam estruturas ou utilizam vetores de peso em algum momento do processo de evolução das soluções candidatas [Asafuddoula et al., 2018, Luque et al., 2020]. Nestes tipos de estudos, os algoritmos propostos tentam de forma progressiva modificar a posição dos vetores de peso, tentando estimar a posição da fronteira Pareto e no final do processo fazer com que todos os subproblemas consigam de fato direcionar alguma solução para uma região Pareto-ótima [Ma et al., 2020]. A atualização dos vetores de pesos não é um

procedimento simples e os algoritmos que implementam este procedimento precisam diferenciar as regiões onde são de difícil aproximação das regiões onde de fato não há soluções ótimas [Qi et al., 2014] e definir a melhor forma de gerar os novos vetores. Também, é importante definir qual o momento mais propício de efetuar a adaptação dos vetores, já que uma atualização prematura poderá ocasionar instabilidade no processo evolutivo e deteriorar as soluções no espaço de objetivos, consequentemente comprometendo o seu desempenho [Giagkiozis et al., 2013]. Outro fator se dá pelo motivo que é difícil realizar um procedimento de adaptação que consiga mapear todos os diferentes tipos de fronteira Pareto e que muitas vezes, utilizar uma estratégia de adaptação poderá afetar a distribuição dos resultados em problemas de fronteira Pareto regulares [Li and Yao, 2020].

Na literatura são apresentados alguns algoritmos que implementam metodologias para a adaptação dos vetores de peso e as abordagens utilizadas para lidar com os eventuais problemas da adaptação. Estudos iniciais, antes mesmo da popularização dos algoritmos baseados em decomposição, propuseram gerar de forma aleatória os vetores de peso a cada nova geração [Ishibuchi and Murata, 1998], sendo uma estratégia que aumenta o custo computacional. Alguns trabalhos apresentam uma abordagem com alocação de recursos dinâmica onde os vetores de peso são fixos, mudando apenas as associações entre as soluções e os vetores. Este é o caso do algoritmo *MOEA/D with Dynamic Resource Allocation* (MOEA/D-DRA) [Zhang et al., 2009], onde é implementado uma função de utilidade para medir a melhora da função de escalarização em cada subproblema em um intervalo de 50 gerações, sendo assim, os recursos são alocados para aqueles que apresentarem as melhores medidas. Os subproblemas são submetidos a uma seleção por torneio e os que apresentarem as melhores medidas são escolhidos para participarem do cruzamento. Adaptando essa abordagem, o *MOEA/D with Generalized Resource Allocation* (MOEA/D-GRA) [Zhou and Zhang, 2015] passa a utilizar uma medida de probabilidade para verificar a melhoria dos subproblemas. Ainda na abordagem fixa, o trabalho proposto por Cai et al. [2015], apresenta o *External Archive Guided MOEA/D* (EAG-MOEA/D) onde é utilizado um arquivo de soluções não-dominadas para auxiliar na evolução. Assim, cada contribuição de um subproblema para o arquivo é verificada e dependendo de quão boas foram, o subproblema é probabilisticamente alocado no processo evolutivo. Apesar destas técnicas serem mais simples e fáceis de implementar, ainda há uma certa dificuldade para garantir uma distribuição dos vetores de pesos nas regiões Pareto, principalmente nas fronteiras de difícil aproximação, uma vez que os vetores de peso não sofreram nenhuma alteração [Li et al., 2015].

A abordagem mais comum dentre os estudos mais recentes, é a alocação de recursos com a atualização geométrica dos vetores de peso. O algoritmo *MOEA/D - with Random Weights* MOEA/D-RW [Li et al., 2015] estende a ideia de geração de vetores de peso de forma aleatória do MOGLS [Ishibuchi and Murata, 1998], porém somente quando a população é considerada estagnada dentro de um determinado número de gerações, me-

lhorando o custo computacional. O *Preference-Inspired co-evolutionary algorithm using Weight Vectors* (PICEA-w) [Wang et al., 2015] propõe co-evoluir os vetores de pesos de forma conjunta com as soluções candidatas durante o processo evolutivo. Neste algoritmo os vetores de peso recebem notas de acordo com a qualidade das soluções encontradas, e assim são ranqueados, onde os melhores são selecionados e os piores são eliminados. Novos vetores são gerados de forma aleatória, compondo assim o conjunto de vetores de peso.

No trabalho de Cheng et al. [2016], é apresentado o *Reference Vector Guided Evolutionary Algorithm* (RVEA). Este algoritmo baseado em decomposição subdivide o espaço de objetivos utilizando os vetores de pesos, gerando assim, subpopulações. Estas subpopulações são determinadas através de um ângulo mínimo que as soluções fazem com um vetor de referência. Uma estratégia de adaptação é implementada com a finalidade de ajustar a distribuição dos vetores de acordo com as escalas dos objetivos. Em Cai et al. [2018], o *Many-objective Evolutionary Algorithm with two Types of Adjustments* (MaOEA-2ADV) é um algoritmo baseado em decomposição que implementa duas estratégias para a adaptação dos vetores de peso. No início, para uma rápida convergência em relação da fronteira Pareto, a busca por soluções é apenas realizada nos vetores limites do espaço de objetivos. Após um número de avaliações e considerando que foi realizada a identificação correta do formato da fronteira na primeira fase, é utilizada uma estratégia baseada em dominância Pareto onde cada solução não-dominada é associada a um vetor de peso e caso o vetor não possua nenhuma associação, este é considerado inefetivo e removido. Novos vetores são inseridos no ponto médio entre dois vetores previamente considerados efetivos.

O *MOEA/D with Adaptive Weight Adjustment* (MOEA/D-AWA) [Qi et al., 2014] explora o ajuste periódico de pesos de forma adaptativa através de uma implementação em duas estratégias com auxílio de um arquivo de soluções não-dominadas. Em um primeiro momento, um conjunto fixo de vetores de pesos é empregado até que seja considerada uma certa convergência das soluções não-dominadas, sendo apenas identificadas que um número de gerações tenha ocorrido. No segundo momento, são adicionados novos subproblemas, provindos do arquivo, em regiões esparsas da população e removido vetores de regiões densas os quais são calculados através de uma medida de esparsidade. No trabalho proposto por Li and Yao [2020] é apresentado o algoritmo MOEA/D-AdaW que também implementa uma metodologia adaptativa dos vetores de peso na estrutura do MOEA/D utilizando o auxílio de um arquivo de soluções não-dominadas. Neste algoritmo, são comparadas a população corrente e a não-dominada com o objetivo de identificar nichos que apresentem regiões subdesenvolvidas e promissoras. Assim, a solução do arquivo é transferida para a população atual e um vetor ótimo àquela solução é calculado com relação a função de agregação de *Tchebycheff* (Equação 2.6). Na operação de remoção, os vetores com os piores valores de função de agregação são removidos até atingir o tamanho populacional  $N$  inicial. O processo de adaptação é inicializado após um determinado número de gerações e mantido fixo a cada 5% do total de gerações. Muito semelhante também é

o MOEA/D-UR [de Farias and Araújo, 2021, de Farias et al., 2018], em que é utilizada uma medida de esparsidade para identificar regiões densas e assim remover e adicionar novos vetores. Para identificar o momento mais propício da adaptação, a variação média da função de escalarização entre gerações é utilizada, ativando a adaptação dos vetores de peso assim que uma baixa variação é identificada.

Outras abordagens complementam os conceitos apresentados previamente inserindo informações da população atual, como por exemplo, verificando a densidade de soluções das regiões próximas aos vetores de pesos, tentando assim, diferenciar as áreas promissoras para adaptação. Os autores do *Nondominated Sorting Genetic Algorithm-III* (NSGA-III) [Deb et al., 2014], algoritmo que utiliza conceitos de fronteiras não-dominadas com a manutenção da diversidade da população através de vetores de referência, apresentaram no trabalho seguinte o *Adaptive NSGA-III* (ANSGA-III) [Jain and Deb, 2014]. O ANSGA-III, incorpora a adaptação do vetor de referência no processo evolutivo. Nesta versão, após o procedimento de contagem de nichos, presente no NSGA-III, para cada vetor de referência com o valor maior que 1 é gerado um conjunto de vetores de pesos através do método do *simplex-lattice* [Das and Dennis, 2000] ao redor do vetor de referência. Na sequência é recalculada a “medida de multidão”, atualizando a contagem de nichos, e os vetores de referência com valor igual a 0 são deletados. No trabalho de Camacho et al. [2019], a metodologia de estimar a densidade e gerar um conjunto de vetores de peso ao redor dos vetores é implementada na arquitetura do MOEA/D [Zhang and Li, 2007], sendo denominado de *Adaptive-MOEA/D* (AMOEAD). É proposta também uma métrica para avaliar a estagnação da população e auxiliar na identificação do momento correto para adaptar os vetores sem interferir no processo evolutivo.

## 2.5 Plataforma PlatEMO

Com o grande aumento das pesquisas relacionadas a algoritmos multi-objetivo vieram também os desafios relacionados a comparações e testes dos algoritmos propostos nestes estudos de maneira eficiente. Assim, com a premissa de desenvolver uma plataforma de pesquisa e ensino, Tian et al. [2017] propuseram uma plataforma para otimização evolucionária multi-objetivo, denominado de PlatEMO.

O PlatEMO é uma plataforma de código aberto e desenvolvida na linguagem MATLAB que incorpora diversos algoritmos multi-objetivo, problemas de teste e métricas para avaliação. Apresenta uma interface gráfica simples que permite com poucos ajustes a comparação entre vários algoritmos, sendo atualmente, disponibilizado com mais de 100 algoritmos. Devido a estas praticidades e a padronização o PlatEMO é utilizado como base nas últimas competições de algoritmos do *IEEE Congress on Evolutionary Computation* (CEC).



## 2.6 Considerações Parciais

Foi apresentada neste capítulo a fundamentação teórica estudada para o desenvolvimento deste trabalho. Foram definidos conceitos introdutórios e termos adotados no estudo de processos de otimização com algoritmos genéticos e realizada uma revisão das principais abordagens na otimização multi-objetivo. Foi realizado um estudo e os principais pontos de melhorias na estrutura conhecida como MOEA/D [Zhang and Li, 2007] e os estudos que apresentam pontos de melhoria, principalmente com foco em adaptação de vetores de pesos. No capítulo seguinte é detalhada a implementação do algoritmo proposto nesta dissertação, o MOEA/D-LNA.

# Capítulo 3

## Algoritmo MOEA/D-LNA

Este capítulo apresenta o algoritmo *Multiobjective Evolutionary Algorithm based on Decomposition with Local Neighborhood-based Adaptation* (MOEA/D-LNA), desenvolvido neste trabalho. O algoritmo é testado utilizando duas técnicas de inicialização de vetores de pesos (Seção 3.2). Na Seção 3.3, é apresentada a técnica de normalização dinâmica do espaço de objetivos que auxilia na evolução e na adaptação dos vetores de pesos. Na Seção 3.4 são discutidos os procedimentos que controlam a frequência e o momento de atualização dos vetores de pesos. Na adaptação dos vetores de pesos é utilizado um conjunto não-dominado de soluções cuja metodologia de sua manutenção é evidenciada na Seção 3.5. Finalmente, na Seção 3.6 são discutidas as estratégias implementadas para a adaptação dos vetores de pesos.

### 3.1 Introdução

O MOEA/D-LNA tem como principais objetivos lidar com problemas de fronteiras irregulares, tentando obter um conjunto de soluções melhor distribuído ao longo de toda fronteira Pareto através da realocação dos vetores de pesos de regiões que não contenham nenhuma solução Pareto ótima para outras regiões promissoras e também ser aplicável à MaOps. Para tal, o algoritmo proposto estende a estrutura do clássico MOEA/D de [Zhang and Li \[2007\]](#) utilizando uma nova estratégia de adaptação dos vetores de pesos na qual as regiões promissoras são identificadas através da análise das  $k$  soluções mais próximas nas vizinhanças dos vetores de peso, estas denominadas de vizinhanças locais. São utilizadas as soluções não dominadas de um arquivo externo em um processo conhecido como adição/remoção, em que novos vetores de pesos são gerados através de um método cônico de geração de vetores de peso.

O pseudocódigo do MOEA/D-LNA é apresentado no Algoritmo 2 e incorpora os fundamentos principais da versão do MOEA/D [[Zhang and Li, 2007](#)] com a função de agregação denominada de *penalty-based boundary intersection* (PBI), com os operadores

genéticos *Simulated Binary Crossover* (SBX) e *Polynomial Mutation* (PMX), em conjunto com o procedimento de seleção e atualização de soluções apresentados na versão denominada de MOEA/D-DE [Zhang et al., 2009] com o parâmetro  $\delta$  de seleção de vizinhança global ou local. O algoritmo utiliza um arquivo contendo uma população não-dominada para orientar a adaptação dos vetores de peso na direção de regiões promissoras do espaço de objetivos. Além disso, para auxiliar no procedimento de adaptação, as seguintes melhorias são também implementadas na estrutura do MOEA/D-LNA:

- Método de inicialização dos vetores de peso alternativo ao presente no MOEA/D Zhang and Li [2007] com o objetivo de gerar um número flexível de vetores de peso em dimensões superiores;
- Normalização dinâmica do espaço de objetivos, em que são consideradas as incertezas dos pontos ideais e nadir nos estágios iniciais da evolução;
- Métrica de melhoria para medir a estagnação da população atual e acionar o procedimento de adaptação conforme necessário;
- Método de manutenção do arquivo para aumentar a qualidade das soluções não-dominadas armazenadas ao longo da evolução.

O procedimento de adaptação dos vetores de peso é dividido em quatro etapas principais. Na primeira etapa, o algoritmo identificará as soluções do arquivo que estão próximas ao conjunto atual de vetores de peso e criará as vizinhanças locais, seguido da remoção dos vetores de peso sem soluções em sua vizinhança local. Na sequência, novos vetores de peso são gerados em regiões promissoras as quais são identificadas através das vizinhanças locais dos vetores remanescentes. Finalmente, os vetores de peso recém-criados são combinados com o conjunto restante e as soluções atuais são associadas aos vetores de peso adaptados. Os detalhes e o procedimento da adaptação dos vetores de peso do MOEA/D-LNA são apresentados nas subseções a seguir.

## 3.2 Inicialização dos Vetores de Pesos

A inicialização do conjunto de vetores de pesos em MOEA baseados em funções de agregação tem um papel de grande importância. Porém, o método implementado no MOEA/D [Zhang and Li, 2007] não permite escalar o número de vetores de pesos para altas dimensões de maneira eficiente e mesmo gerando uma distribuição uniforme inicial dos vetores, não há garantias de um resultado também uniformemente distribuído.

Neste caso, para verificar a eficiência do método de adaptação de vetores de peso proposto no MOEA/D-LNA, são investigados dois métodos para inicialização de vetores

**Algorithm 2** MOEA/D-LNA**Input:**  $N, T_{\max}, t_{\text{on}}, t_{\text{off}}, nr$ 


---

```

1: Inicializa a população  $P(t = 0)$ ;
2: Inicializa os vetores de peso  $W(t = 0)$  (Seção 3.2);
3: Calcula as vizinhanças  $B(i)$  dos vetores de peso  $w_i \in W(t)$ ;
4: Inicializa o arquivo  $A$  com as soluções não dominadas  $P$ ;
5: while  $t < T_{\max}$  do
6:   Atualiza as estimativas dos pontos ideal  $z^*$  e nadir  $z^\circ$ ;
7:   Aplica a normalização dinâmica (Seção 3.3);
8:   for cada sub-problema  $i = 1 : N$  do
9:     if  $U(0, 1) \leq \delta$  then  $E \leftarrow B(i)$  else  $E \leftarrow \{1, \dots, N\} \setminus \{i\}$ 
10:     $p \leftarrow$  Gera uma nova solução a partir do conjunto de reprodução  $E$ ;
11:    if  $g(p|w_i, z^*) \leq g(x_i|w_i, z^*)$  then
12:       $x_i \leftarrow p$ ;
13:    end if
14:     $cr \leftarrow 0$ 
15:    for cada  $j \in E$  do
16:      if  $g(p|w_j, z^*) \leq g(x_j|w_j, z^*)$  and  $(cr < nr)$  then
17:         $x_j \leftarrow p$ ;  $cr \leftarrow cr + 1$ ;
18:      end if
19:    end for
20:  end for
21:   $\tilde{D} \leftarrow$  Calcula métrica de melhoria (Seção 3.4);
22:  Atualiza o arquivo de soluções não-dominadas  $A$  (Seção 3.5);
23:  if  $(t_{\text{on}} \leq t \leq t_{\text{off}})$  and  $(\tilde{D} \leq 0,001)$  then
24:     $W(t) \leftarrow$  Atualiza vetores de peso (Seção 3.6);
25:    Recalcula as vizinhanças  $B(i) \forall w_i \in W(t)$ ;
26:  end if
27: end while
28: return  $A$ 

```

---

de peso: o  $k$ -layers [Jiang and Yang, 2017] e o *Riesz S-Energy* [Blank et al., 2020]. Como evidenciado na Seção 2.3.1, os dois métodos escolhidos possuem abordagens diferentes porém apresentam controle mais flexível do número de vetores de peso quando se comparado ao método original. Aliado a esse controle do número de vetores, a ideia central aqui, é verificar se independente da técnica de inicialização o algoritmo é capaz de ajustar os vetores de pesos.

### 3.3 Normalização Dinâmica dos Objetivos

Um aspecto importante dos problemas de otimização são as diferenças na escala dos objetivos. Comumente, os objetivos estão em medidas distintas e podem afetar a evolução, alterando a direção da busca e comprometendo os resultados finais, principalmente quando se trata de problemas com muitos objetivos. Essa dificuldade pode ser amenizada com o uso de técnicas de normalização, não só melhorando o processo de evolução, mas

também gerando um conjunto de soluções com maior diversidade no espaço de objetivos. Além disso, um espaço de objetivos normalizado é importante na atualização dos vetores de peso, uma vez que as distâncias entre as soluções e os vetores de peso são calculadas neste processo.

Uma das formas mais comuns para normalizar o espaço de objetivos em problemas de dimensão  $M$ , consiste em utilizar os pontos utópico  $z^* = (z_1^*, \dots, z_M^*)'$  e nadir  $z^\circ = (z_1^\circ, \dots, z_M^\circ)'$ . O ponto utópico pode ser obtido minimizando cada um dos objetivos de forma individual, já o ponto nadir precisa ser identificado da fronteira Pareto. Para facilitar o processo de normalização, esses pontos são geralmente estimados como sendo o ponto mínimo e máximo de cada geração. Após a identificação, a normalização de  $f_i$  para o  $i$ -ésimo objetivo é realizada através da equação (3.1):

$$f_i^l = \frac{f_i - z_i^l}{z_i^u - z_i^l + \epsilon} \quad (3.1)$$

em que os pontos  $z_i^l$  e  $z_i^u$  são os respectivos valores estimados dos pontos utópico e nadir para cada objetivo. Já o  $\epsilon$  é um valor positivo que tem por objetivo prevenir o denominador de se tornar zero. Após a normalização, os objetivos são redimensionados, ficando entre os limites de  $[0,1]$ .

Um dos desafios desse procedimento é estimar adequadamente os pontos extremos. A estimativa do ponto utópico é bastante simples, a cada geração será o valor mínimo de cada objetivo. Portanto, ele é atualizado da seguinte forma:

$$z_i^l = \min(z_i^l \cup P_i) \quad (3.2)$$

onde  $P_i$  contém os valores do  $i$ -ésimo objetivo para a população atual. Por outro lado, a estimativa do ponto nadir não é uma tarefa fácil, pois não pode ser atualizada maximizando os objetivos. O pior ponto alcançado em um problema é diferente do ponto nadir, pois depende da fronteira Pareto a ser definida [Deb et al., 2010]. Consequentemente, o ponto nadir é frequentemente estimado como o valor máximo de cada objetivo considerando a estimativa atual da fronteira Pareto. Portanto, tanto o ponto ideal quanto o nadir são constantemente estimados e atualizados durante a evolução.

Nas etapas iniciais do processo de evolução, existem muitas incertezas que podem comprometer as estimativas dos pontos extremos. Além disso, como os pontos estão em constante atualização e o processo de normalização ocorre antes da associação do vetor de peso e avaliação da função de decomposição, uma variação significativa pode resultar em uma degradação e instabilidade durante o processo de evolução [Blank et al., 2019]. Alguns trabalhos propuseram abordagens alternativas para lidar com estimativa de pontos extremos e normalização de objetivos. Por exemplo, no caso dos algoritmos  $\theta$ -DEA [Yuan et al., 2015] e o NSGA-III [Deb et al., 2014], foi usado um método de normalização

baseado em hiperplano. Nesta abordagem, um hiperplano é construído com os  $M$  pontos extremos da população e o ponto nadir estimado é identificado como o ponto que o intercepta. Embora este e alguns outros métodos alternativos sejam mais refinados, um estudo recente [He et al., 2020] propôs um simplificado método de normalização dinâmica, no qual é possível controlar a intensidade da normalização. Nessa abordagem, a equação (3.1) é alterada para a equação (3.3) e a equação (3.4), onde  $\alpha$  é parâmetro responsável pelo controle da normalização:

$$f'_i = \frac{f_i - z_i^l}{L_i + \epsilon} \quad (3.3)$$

$$L_i = \frac{z_i^u - z_i^l}{(1 - \alpha)(z_i^u - z_i^l - 1) + 1} \quad (3.4)$$

A ideia por trás do método de normalização dinâmica é que nos estágios iniciais da evolução, quando há pouca ou nenhuma informação confiável a respeito da fronteira Pareto,  $\alpha = 0$ , desacoplando a equação (3.3) da estimativa do ponto nadir. Conforme a evolução avança e uma melhor estimativa do ponto nadir pode ser alcançada com a convergência,  $\alpha$  aumenta, permitindo uma maior influência do ponto nadir no processo de normalização. Finalmente, quando  $\alpha = 1$ , equação (3.3) torna-se igual a (3.1). No artigo original, os autores apresentam duas funções para controlar o parâmetro  $\alpha$ , uma usando uma função linear e a outra uma função sigmoide. Neste trabalho, a versão com a função sigmoide é implementada:

$$\alpha(t) = \frac{1}{1 + \exp \left[ -8 \left( \frac{t - 1}{T_{max} - 1} - 0.5 \right) \right]} \quad (3.5)$$

onde  $t$  é a geração atual e  $T_{max}$  é o número total de gerações. Neste trabalho, o conceito de número total de gerações é alterado para o número mínimo de gerações antes que o algoritmo possa atualizar os vetores de peso pela primeira vez. Com isso, pretende-se que o espaço de objetivos seja totalmente normalizado antes da primeira adaptação dos vetores de peso, melhorando a estimativa dos pontos extremos e garantindo a normalização do espaço de objetivos para o processo de adaptação.

Outro ponto que também foi verificado por Ishibuchi et al. [2017a] é a influência do parâmetro  $\epsilon$  no procedimento de normalização apresentado na equação (3.1). Quando  $\epsilon = 10^{-6}$ , ou seja, um valor muito pequeno, foi observada uma deterioração dos resultados em problemas com escalas diferentes. Definindo este parâmetro como  $\epsilon = 1$ , melhores resultados nestes tipos de problemas foram encontrados. As diferenças nos resultados pela alteração do parâmetro  $\epsilon$  são devidas à perda de diversidade ao realizar a normalização dos objetivos com diferentes escalas. Ao analisar a diferença entre os pontos estimados nadir e o ponto utópico,  $(z_i^u - z_i^l)$ , é possível detectar uma variação entre os objetivos, e

quando  $\epsilon$  é muito pequeno, o espaço do objetivo tende a ser direcionado mais para um dos objetivos. Na Figura 3, é possível observar a influência do  $\epsilon$  nos intervalos do problema DTLZ7 com 5 objetivos. Na Figura 3a, 4 intervalos de objetivo estão mais próximos de zero e isso pode influenciar a normalização do espaço de objetivo em direção ao 5º objetivo, de maior valor. Na Figura 3b por outro lado, com  $\epsilon = 1$ , as diferenças dos pontos superior e inferior estão mais próximas de 1.

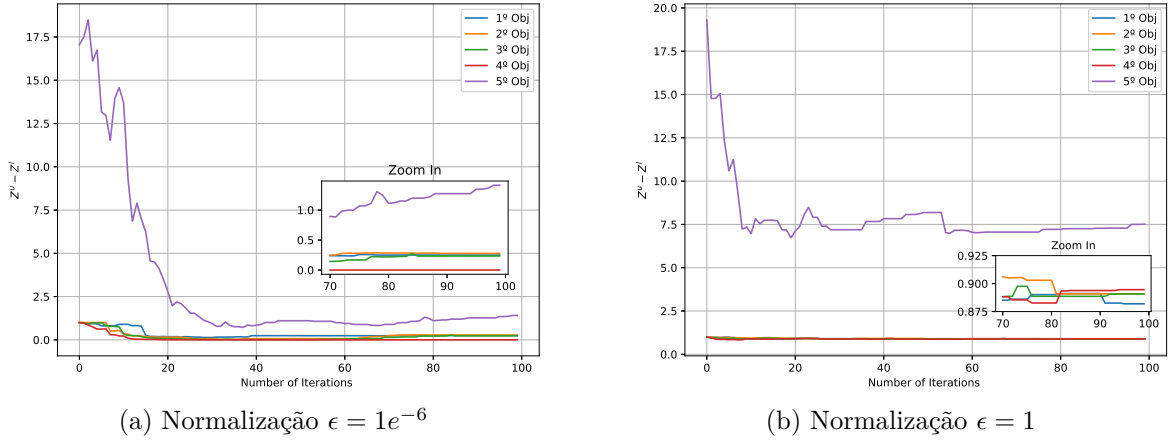


Figura 3 – Diferença do intervalo das soluções para cada objetivo para a função de benchmark DTLZ7 (Fonte: Própria).

Embora o procedimento de normalização com  $\epsilon = 1$  mostre um melhor desempenho em problemas com objetivos com diferentes escalas, o oposto é notado em problemas de teste onde a normalização não é necessária. Para contornar esses problemas durante o procedimento de normalização, é proposto um ajuste dinâmico do parâmetro  $\epsilon$ , realizado conforme abaixo:

$$\epsilon(t) = \begin{cases} 10^{-6}, & \text{if } |\Delta_i(t) - \Delta_j(t)| \leq \zeta, \forall i, j \\ 1, & \text{Caso contrário} \end{cases} \quad (3.6)$$

com  $\Delta_i = z_i^u - z_i^l$ . Ao fazer isso, o método identifica qualquer mínima diferença maior que um limiar,  $\zeta$ , nas escalas de objetivos e, em seguida, altera o parâmetro  $\epsilon$  de acordo.

### 3.4 Frequência de Atualização dos Vetores de Peso

É comum na literatura realizar o procedimento de atualização dos vetores de pesos a cada determinado número de gerações, geralmente definida como uma porcentagem do total de avaliações como no trabalho de Li and Yao [2020]. Além disso, alguns outros estudos, por exemplo em [Qi et al., 2014], até mesmo é adicionado um número mínimo de gerações antes do início desta atualização periódica.

Atualizar os vetores de peso em um problema multi-objetivo não é uma tarefa trivial. No início da evolução, existem muitas soluções dominadas no espaço de objetivos, incertezas na hora de estimar os pontos extremos, como discutido na seção anterior, e a possibilidade de desencadear a convergência desigual das soluções [Giagkiozis et al., 2013]. Assim, uma atualização prematura dos vetores de peso pode afetar de forma direta o processo de otimização, deteriorando o espaço de objetivos e a busca de soluções ótimas. Ainda, após a atualização dos vetores de pesos, as soluções no espaço de objetivos serão reajustadas de acordo com os novos vetores de peso, e este processo poderá levar um determinado número de gerações até que estabilize.

Para amenizar esses problemas, no MOEA/D-LNA, dois critérios devem ser atendidos para permitir o início do processo de atualização. Como muitos outros métodos que adaptam os vetores de peso, o primeiro critério é um número mínimo de gerações desde o início da evolução, aqui denominado de  $t_{\text{on}}$ . No MOEA/D-LNA o tempo mínimo é importante para deixar o algoritmo convergir adequadamente e para melhor estimar os pontos extremos que são usados no processo de normalização dinâmica. Ao final deste período, as soluções estarão totalmente normalizadas através da equação (3.3) com  $\alpha = 1$ . Para evitar uma degradação das soluções no final da evolução, que pode ocorrer devido ao tempo insuficiente para as soluções serem reajustadas, o processo de atualização é desativado quando o número de avaliações atinge 90% do total. Portanto, as atualizações dos vetores de peso são permitidas apenas quando dentro do intervalo  $t_{\text{on}} \leq t \leq t_{\text{off}}$ , com  $t_{\text{off}} = 0,9T_{\text{max}}$ .

Na segunda condição, para evitar uma possível situação em que uma atualização possa ser acionada quando as soluções ainda não estão estabilizadas no espaço de objetivos, uma métrica é utilizada para medir a variação do progresso das soluções durante a evolução nas últimas  $\Delta t$  gerações. Em [Camacho et al., 2019], os autores apresentam uma interessante métrica de melhoria denominada de Dispersão Mediana de População. Para medir a melhoria das soluções, a equação (3.7):

$$u_j(t) = \frac{g(x_j(t - \Delta t)) - g(x_j(t))}{g(x_j(t - \Delta t))} \quad (3.7)$$

é usada para calcular a melhoria relativa na função de agregação,  $g$ , de cada  $j$ -ésimo subproblema nas últimas  $\Delta t$  gerações. Um valor de melhoria relativa próximo de zero significa uma estagnação da solução  $x_j$ . Para aumentar a qualidade da métrica e o grau de confiança da variação da população, são salvas as últimos  $l$  melhorias relativas de cada subproblema.

O coeficiente de dispersão  $D_j$  do intervalo de tempo  $u_j(t), \dots, u_j(t - l)$  é então definido como a razão da variância para a média. Portanto, a métrica de melhoria de toda a população é calculada como a mediana dos coeficientes de dispersão,  $\tilde{D}$ . Um valor resultante inferior a um certo liminar, próximo a zero, indicará que a população estagnou,



ou seja não há nenhuma melhora considerável nos objetivos. Neste trabalho, os parâmetros usados foram  $\Delta t = 10$ ,  $l = 5$  e o limiar é igual a 0,001.

### 3.5 Manutenção do Arquivo de Soluções não Dominadas

No MOEA/D-LNA, o arquivo é um aspecto importante do método, pois é usado para orientar a adaptação dos vetores de peso em direção a regiões promissoras do espaço objetivo. O arquivo é predefinido com o mesmo tamanho da população,  $N$ , e armazena as soluções não dominadas encontradas durante o processo evolutivo. Quando a capacidade do arquivo excede seu limite, um mecanismo de manutenção de diversidade é executado, pois é importante preservar soluções representativas que guiam a adaptação dos vetores de peso. Para tanto, é implementado o método de manutenção baseado em nicho apresentado em [Li et al., 2016].

Na abordagem, a região das soluções armazenadas é considerada para medir a aglomeração das soluções contidas no arquivo e é também usada para remover as piores soluções, sendo aquelas em regiões mais densas, até que o limite máximo de capacidade seja atingido novamente. Considerando uma solução  $p$  no arquivo não dominado, a estimativa de densidade é calculada da seguinte forma:

$$D(p) = 1 - \prod R(p, q) \quad (3.8)$$

$$R(p, q) = \begin{cases} d(p, q)/r, & \text{if } d(p, q) \leq r \\ 1, & \text{Caso contrário} \end{cases} \quad (3.9)$$

onde  $d(p, q)$  é uma medida de distância entre as soluções  $p$  e  $q$  e o parâmetro  $r$  é o raio do nicho que é definido como a média do  $n$ -ésimo vizinho mais próximo da solução verificada. No artigo original [Li et al., 2016], a métrica de distância usada é a Euclidiana e o tamanho máximo da vizinhança é  $n = 3$ . Em [Li and Yao, 2020] a mesma abordagem é implementada, mas com  $n = M$  e a mediana em vez da média. Essas alterações reduzem a parametrização e apresentam melhores resultados em dimensões superiores, portanto, também é incorporado no MOEA/D-LNA.

### 3.6 Estratégia de Atualização dos Vetores de Pesos

No procedimento de atualização do MOEA/D-LNA, objetiva-se remover os vetores de peso que não contêm nenhuma parte da fronteira Pareto e gerá-los em regiões promissoras que são identificadas pelas soluções não dominadas armazenadas no arquivo.

Também, neste processo, busca-se equilibrar a adição de novos vetores de peso entre regiões que apresentam muitas soluções por se tratarem das bordas da fronteira Pareto e outras regiões que são promissoras e pouco exploradas. Portanto, após os critérios de atualização, explicados na seção 3.4, terem sido atendidos, o algoritmo iniciará o procedimento de atualização dos vetores de peso. Este procedimento é dividido em quatro etapas principais:

1. Definição das vizinhanças locais pelo cálculo do número de soluções no arquivo que estão próximas de cada vetor de peso atual;
2. Remoção de vetores de peso que não têm solução em sua vizinhança local;
3. Geração de um novo conjunto de vetores de peso com base nestas vizinhanças locais;
4. Combinação de conjuntos de vetores de peso e associação da população atual.

### 3.6.1 Definição das Vizinhanças Locais

Na primeira etapa principal do procedimento responsável pela adaptação dos vetores de peso, as soluções armazenadas no arquivo não dominado são utilizadas para identificar as áreas promissoras encontradas até agora durante a evolução. Portanto, calcula-se quantas soluções do arquivo estão próximas de cada vetor de peso  $w_i \in W$ , constituindo assim as vizinhanças locais  $L(i)$ .

Para exemplificar este procedimento, a Figura 4 representa um cenário após  $t_{on}$  gerações, em um problema com uma fronteira Pareto irregular e cinco soluções do arquivo não-dominado. No exemplo proposto, as vizinhanças locais para os vetores de peso  $w_1$ ,  $w_4$  e  $w_5$  serão compostas por apenas uma solução cada,  $\{s_1\}$ ,  $\{s_4\}$  e  $\{s_5\}$ , respectivamente. No entanto, para o vetor de peso  $w_2$ , não há soluções próximas, pois o ângulo entre a solução  $s_2$  e o vetor de peso  $w_3$ ,  $\theta_{2,3}$  é menor que o ângulo entre  $s_2$  e o vetor de peso  $w_2$ ,  $\theta_{2,2}$ . Portanto, as vizinhanças locais conterão  $\{\emptyset\}$  e  $\{s_2, s_3\}$ , para  $w_2$  e  $w_3$ , respectivamente.

No Algoritmo 3 o pseudocódigo do procedimento para calcular as vizinhanças locais é apresentado. Nas linhas (1-3), as vizinhanças locais são inicializadas e um conjunto auxiliar  $S$  para armazenar soluções que não entraram em nenhuma vizinhança é criado. A matriz de distância entre as soluções do arquivo para todos os vetores de peso é calculada. A matriz de distância é calculada pela distância angular, dada por:

$$\theta_{i,j} = \arccos \left( \frac{s_i \cdot w_j}{\|s_i\| \times \|w_j\|} \right) \quad (3.10)$$

Na sequência, nas linhas (4-11) para cada solução, o  $j$ -ésimo vetor de peso mais próximo é identificado através da matriz de distância e se a vizinhança local deste vetor

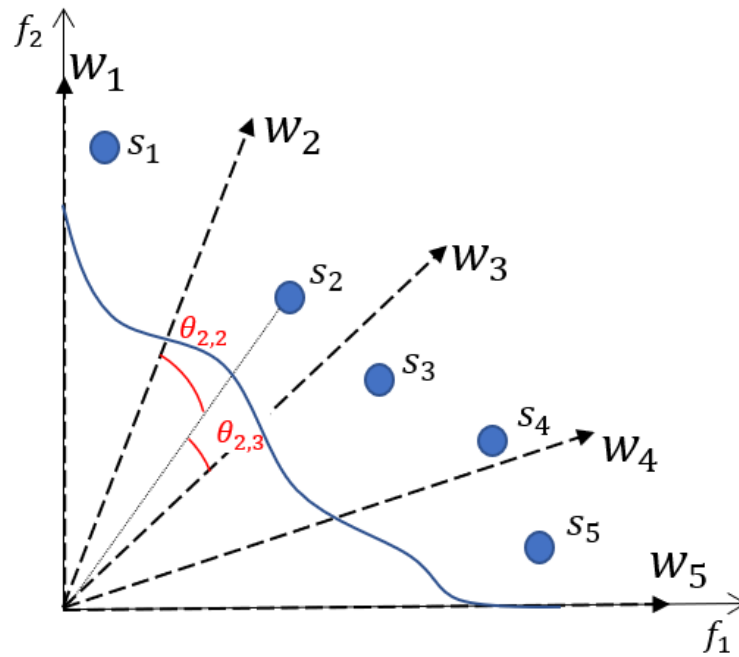


Figura 4 – Exemplo da construção das vizinhanças locais. Na figura, os pontos azuis representam soluções no arquivo, as setas tracejadas são os vetores de peso e a fronteira Pareto é representada como uma linha sólida azul (Fonte: Própria).

de peso ainda não atingiu a capacidade máxima, a solução é então atribuída ao vetor. Para evitar que muitas soluções sejam alocadas a uma única região e para aumentar a exploração, é definida uma limitação de  $k$  soluções por vizinhança local. Então, se uma solução for identificada como tendo o menor ângulo com um vetor de peso cuja vizinhança local atingiu a capacidade máxima, a solução e o índice de vetor mais próximo serão armazenados no conjunto temporário  $S$ .

Nas linhas (12-15), para controlar o número de vetores de peso alterados a cada execução do processo de adaptação, é calculado o número de vizinhanças locais vazias. Se o número total de vetores de peso removidos for menor que o parâmetro  $\delta_c$ , as soluções do conjunto temporário  $S$  serão alocadas aleatoriamente para vizinhanças locais que estão vazias. Ao fazer isso, os vetores de peso que seriam excluídos são mantidos na adaptação atual.

Finalmente, quaisquer soluções que ainda permaneçam no conjunto  $S$  serão redistribuídas, linhas (16-30). Com isso, pretende-se realocar soluções  $s_i \in S$  para as menores vizinhanças locais, cujo tamanho é diferente de zero, aumentando a exploração e evitando a criação de regiões aglomeradas. Portanto, com a probabilidade  $\delta_r$ , uma solução  $s_i$  será atribuída aleatoriamente a um vetor em  $B(j)$  do  $w_j$  anterior que possui a menor vizinhança local até o dado momento. Caso contrário,  $s_i$  será atribuída aleatoriamente a qualquer vizinhança local com o menor tamanho. Ao final deste procedimento, todas as

vizinhanças locais dos vetores de peso atuais serão criadas, tendo de zero a  $k$  soluções e uma solução sendo atribuída a apenas uma única vizinhança local.

---

**Algorithm 3** Cálculo das Vizinhanças Locais
 

---

**Input:**  $W, A, \delta_r, \delta_c$

```

1: Inicializa as vizinhanças locais  $L$ ;
2: Inicializa o conjunto temporário  $S$ ;
3: Calcula a distância angular  $\theta_{i,j}$  entre  $a_i \in A$  e  $w_j \in W$ ;
4: for para cada solução no arquivo  $a_i \in A$  do
5:   Encontra o vetor de peso mais próximo ( $w_j$ ) da solução  $a_i$ ;
6:   if  $|L(j)| < k$  then adiciona  $a_i$  a  $L(j)$  else adiciona  $a_i$  a  $S$ 
7: end for
8:  $pw \leftarrow$  Calcula a porcentagem de vetores em  $L = \emptyset$ ;
9: while ( $pw > \delta_c$ ) do
10:  Aleatoriamente redistribui as soluções de  $S$  para alguma vizinhança vazia  $L(j)$ ;
11: end while
12: if ( $S$  não  $\emptyset$ ) then
13:  for toda  $s_i \in S$  do
14:     $j \leftarrow$  Encontra  $L(\cdot)$  com menor tamanho;
15:    if  $U(0, 1) < \delta_r$  then
16:      Aleatoriamente escolhe  $w_c$  de  $B(j)$ 
17:    else
18:      Aleatoriamente escolhe  $w_c \in W$  tal que  $|L(c)| > 0$ ;
19:    end if
20:     $L(c) \leftarrow s_i$ ;
21:  end for
22: end if
23: return  $L$ 

```

---

### 3.6.2 Remoção dos Vetores de Peso

Na segunda etapa, as informações de densidade obtidas anteriormente são utilizadas para inferir sobre a extensão das regiões da fronteira Pareto encontradas até o momento. Além disso, as soluções excedentes redistribuídas do conjunto  $S$  serão importantes para aumentar os tamanhos das vizinhanças locais e, conseqüentemente, aumentar o número de vetores de peso em regiões promissoras e pouco exploradas que serão gerados nas etapas seguintes. Portanto, o número de soluções em cada vizinhança local  $L(j)$  é calculado e todos os vetores de peso com uma vizinhança vazia serão deletados, as soluções restantes irão gerar o subconjunto de vetores de peso denominado de  $W'$ .

### 3.6.3 Adição de Novos Vetores de Peso

Dado o subconjunto dos vetores de peso  $W'$  e as informações de vizinhanças locais obtidas na etapa anterior, é realizada a adição dos novos vetores de peso nas regiões promissoras. Para criar os novos vetores de peso no MOEA/D-LNA, é adaptada uma versão do método de geração de vetores de peso, *Cone Weight Vector Generator* (WVG)

proposto por [Meneghini et al. \[2020b\]](#). O WVG é um método gerador de vetores de peso que implementa um algoritmo evolutivo de estado estacionário e que gera um conjunto de vetores de peso dentro de um cone de preferência definido por um vetor de direção de preferência  $v$  e um ângulo de abertura  $\xi$ . O método foi aplicado para otimização baseada em preferências em [\[Ferreira et al., 2020\]](#) e [\[Meneghini et al., 2021\]](#).

O método WVG é semelhante aos algoritmos usados para gerar vetores de peso em MOEAs baseados em funções de agregação. A ideia por trás do método, é evoluir um conjunto aleatório inicial de vetores de peso,  $N$ , localizados em um hipercubo definido por  $[0, 1]^M$ . Primeiro, é inicializada aleatoriamente uma população de vetores de peso  $N$  e normalizada com a norma  $p$ , onde  $p = 1$ . Em seguida, a matriz de distância entre cada vetor no conjunto inicial é calculada. Logo após, um novo descendente de cada vetor é criado através da adição de uma perturbação Gaussiana ao vetor original, também normalizado com a norma  $p$ , e calculada a distância do novo vetor à população atual. A função de aptidão será a soma dos  $T$  vizinhos mais próximos do conjunto original  $W$  mais o parâmetro de penalidade definido pelo ângulo de abertura  $\xi$  entre o novo vetor e o vetor de preferência  $v$ . Finalmente, o pior vetor na população atual é substituído pelo descendente recém-criado, caso haja melhora na função de aptidão. Após o número definido de gerações, o algoritmo termina com uma população de vetores de peso dentro do cone definido.

Na Figura 5, um exemplo é apresentado em que um conjunto de vetores de peso foi gerado em um espaço tridimensional através do método de geração do *simplex-lattice*. Em seguida, são executadas três instâncias do método cone WVG usando três vetores selecionados aleatoriamente do simplex como os vetores de preferências, com três ângulos de abertura,  $\xi$ , distintos e tamanho de população  $N = 20$ . É possível observar como os cones dos vetores de peso são gerados no espaço e como o ângulo de abertura influencia na distribuição das soluções.

Devido às propriedades de geração de um número variável de vetores de peso dentro dos cones e controle de ângulo flexível, o cone WVG torna-se um método interessante e é aplicado no MOEA/D-LNA, especificamente na etapa de adição de vetores de peso. Assim, o Algoritmo 4 ilustra o pseudocódigo do procedimento de criação de vetores de peso onde é gerada uma instância do cone WVG para cada vetor de peso  $w_i \in W'$  que tem o tamanho da vizinhança local maior que 1, originando o novo conjunto  $W_g$ .

Primeiro, é calculado o ângulo de abertura  $\xi$  dado pelo ângulo da hiper-diagonal multiplicado pela variável  $\phi$  como na equação (3.11) (linha 2):

$$\xi = \arccos\left(\frac{1}{\sqrt{M}}\right) \times \phi \quad (3.11)$$

Durante o desenvolvimento, verificou-se que o ângulo de abertura precisa ser ajustado de acordo com o aumento da dimensão, portanto, é proposto que  $\phi$  seja calculado

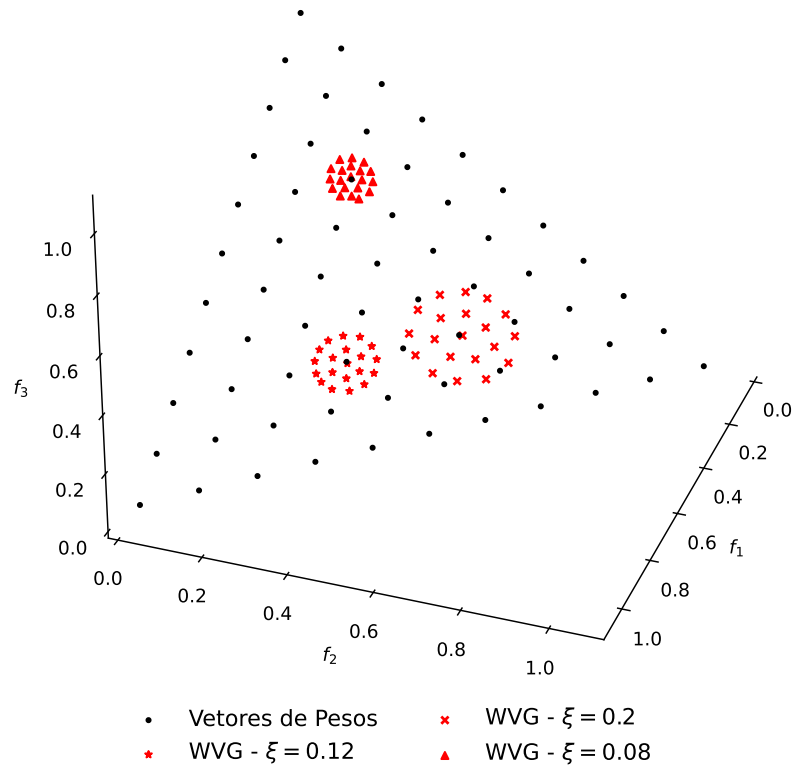


Figura 5 – Exemplo de três diferentes cones gerados pelo método WVG em um espaço tridimensional.

---

**Algorithm 4** Gerando Novos Vetores de Peso

---

**Input:**  $W', L(i), c$

- 1: Inicializa o conjunto vazio  $W_g$ ;
  - 2:  $\xi \leftarrow$  Calcula o ângulo de abertura;
  - 3: **for** para cada vetor de peso  $w_i \in W'$  tal que  $|L(i)| > 1$  **do**
  - 4:     $v \leftarrow w_i$ ;
  - 5:     $N_{g,i} \leftarrow c \times |L(i)|$ ;
  - 6:     $W_g \leftarrow W_g \cup \text{WVG}(N_{g,i}, v, \xi)$ ;
  - 7: **end for**
  - 8: **return**  $W_g$
- 

como uma função linear <sup>1</sup> que varia com a dimensão  $M$  do problema.

O vetor de preferência é definido como o vetor de peso  $w_i$  (linha 4) e o tamanho da população do cone de vetores de peso gerado é o mesmo que a quantidade de soluções em cada vizinhança local  $L(i)$  multiplicado por uma constante  $c$ , definido como  $N_{g,i}$  (linha 5). O parâmetro  $c$  é um inteiro positivo que funciona como forma de aumentar a diversidade quando as vizinhanças locais são pequenas, por exemplo, se  $c = 1$  o cone gerado terá tantos vetores de peso quanto o tamanho da vizinhança local de seu vetor de preferência.

---

<sup>1</sup>  $\phi = -0,0322 + 0,0353 \times M$

Finalmente, é gerado um cone WVG para  $i$ -ésimo vetor de peso e este é adicionado ao conjunto  $W_g$ . O conjunto de vetores  $W_g$  conterá todos os novos vetores gerados para cada vizinhança.

### 3.6.4 Associação dos Vetores de Peso

Ao final da etapa anterior haverão dois conjuntos de vetores de peso, o conjunto  $W'$  contendo o restante dos vetores do conjunto inicial, com vizinhanças locais maiores que zero e o conjunto recém-criado  $W_g$  contendo todos os vetores de peso gerados através do método cone WVG.

Na quarta e última parte do procedimento de adaptação do MOEA/D-LNA, os conjuntos  $W'$  e  $W_g$  são combinados para originar o conjunto adaptado de vetores de peso. Para chegar ao tamanho inicial da população,  $N$ , é implementada uma ideia que é geralmente utilizada para inicializar vetores de peso, conhecida por Método de Construção [Deb et al., 2019].

Esses procedimentos geralmente começam com um conjunto grande de pontos e, na sequência, novos pontos são adicionados um por vez até que o conjunto atinja o tamanho necessário. Os novos pontos adicionados são escolhidos por uma métrica que mensura algum tipo de diversidade ou densidade. Por exemplo, em [Zhang et al., 2009], são tomados os pontos que maximizam a distância Euclidiana para o conjunto final.

Motivado por esses métodos de inicialização, foi implementado um procedimento para adicionar os vetores de peso recém-criados de  $W_g$  a  $W'$ , cujo pseudocódigo é apresentado no Algoritmo 5.

---

#### Algorithm 5 Construção do Conjunto Final de Vetores de Peso

---

**Input:**  $W', W_g, N$

- 1: **while**  $|W'| < N$  **do**
  - 2:   Encontra o  $W_g(i)$  com a maior distância entre  $W'$  e  $W_g$ ;
  - 3:   Adiciona  $W_g(i)$  a  $W'$ ;
  - 4:   Remove  $W_g(i)$  de  $W_g$ ;
  - 5: **end while**
  - 6: **return**  $W'$
- 

Essencialmente, é identificado o vetor de peso  $W_g(i) \in W_g$  que maximiza as distâncias angulares de todos os vetores peso ao conjunto de vetores restante  $W'$  (linha 2). Depois disso,  $W_g(i)$  é excluído de  $W_g$  e transferido para o conjunto  $W'$  (linhas 3-4). Este procedimento é repetido até que o tamanho do conjunto  $W'$  atinja o tamanho da população atual,  $N$ , e então o procedimento termina. Ao utilizar esta abordagem, o MOEA/D-LNA visa complementar uniformemente o conjunto final com os vetores de peso promissores gerados no conjunto  $W_g$ . Por último, são calculadas as distâncias angulares entre o conjunto de vetores de peso final em  $W'$  e as soluções da população atual e

cada solução é associada ao vetor de peso mais próximo, finalizando assim, o processo de atualização de vetores de peso.

### 3.7 Considerações Parciais

Este capítulo apresentou o modelo do algoritmo proposto, o MOEAD/D-LNA, iniciando com a visão geral da estrutura e dos componentes principais que o difere do MOEA/D [Zhang and Li, 2007]. Na sequência, foram especificadas as melhorias implementadas na estrutura assim como suas razões. Finalmente, foi delineado todo o procedimento para realizar o processo de adaptação dos vetores de pesos.

No capítulo seguinte, são realizadas análises dos parâmetros relacionados ao procedimento de adaptação de vetores de peso, os experimentos com funções de teste e comparações com outros algoritmos.



# Capítulo 4

## Resultados Experimentais

Este capítulo tem por finalidade comparar o MOEA/D-LNA com outros algoritmos evolucionários estado da arte e os que também propõem algum tipo de adaptação dos vetores de pesos sob a perspectiva de problemas de teste com avaliações quantitativas e qualitativas.

Na Seção 4.1 são detalhadas as configurações e plataformas utilizadas nos experimentos. Os problemas para validar o algoritmo e suas configurações são apresentados na Seção 4.2. Na Seção 4.3 são analisados os parâmetros que fazem parte do MOEA/D-LNA. Na sequência são apresentados os resultados e as discussões dos experimentos computacionais na Seção 4.4.

### 4.1 Configuração Experimental

O MOEA/D-LNA foi implementado em MATLAB na plataforma PlatEMO [Tian et al., 2017], versão 2.9, em que os algoritmos para comparação estavam disponibilizados. A máquina em que foram executados todos os experimentos possui 16 gigabytes de memória RAM e um processador Intel Core i7-4500U de 1.80GHz de 2 núcleos e 4 Threads.

Para a validação do MOEA/D-LNA, foram feitas comparações entre outros 11 MOEA da literatura, os quais foram divididos em dois grupos. No grupo 1, estão os MOEA que não implementam nenhuma estratégia de adaptação de vetores de peso, sendo estes: NSGA-III [Deb et al., 2014], MOEA/DD [Li et al., 2015], MOEA/D-DE [Li and Zhang, 2009], MOEAD-M2M [Liu et al., 2014] e o MOEA/D [Zhang and Li, 2007] com PBI. Em contrapartida, no grupo 2, estão os algoritmos que implementam alguma das estratégias apresentadas no Capítulo 2: ANSGA-III [Jain and Deb, 2014], RVEA [Cheng et al., 2016], MOEA/D-AWA [Qi et al., 2014], EAG-MOEA/D [Cai et al., 2015] e MOEA/D-DRA [Zhang et al., 2009]. Os parâmetros de cada competidor foram ajustados conforme já disponibilizado no PlatEMO e os seus artigos originais. Para também verificar a efetividade da estratégia de adaptação dos vetores de peso, o MOEA/D-LNA é comparado com sua

versão sem o procedimento de adaptação (etapa 3.6), nomeada aqui de MOEA/D-LNA\*. Por sua vez, para o MOEA/D-LNA foram considerados os seguintes parâmetros:

- Probabilidade de cruzamento:  $p_c = 1$  e probabilidade de mutação:  $p_m = 1/d$
- Tamanho da vizinhança:  $T = N/10$
- Probabilidade de compartilhamento da vizinhança  $\delta = 0.9$
- Número máximo de soluções substituídas na vizinhança  $nr = 2$
- Parâmetro de penalidade da PBI:  $\theta = 5$
- Número de gerações/avaliações:  $T_{Max}$ , a depender do problema e da dimensão
- Número mínimo e máximo de gerações/avaliações em que é permitido atualizar os vetores:  $t_{on} = 0,25T_{Max}$  e  $t_{off} = 0,9T_{Max}$
- Fator de multiplicação de vetores gerados no cone WVG:  $c = 2$
- Porcentagem máxima de vetores que podem ser alterados  $\delta_c = 0,3$
- Probabilidade da solução em  $s_i \in S$  ser alocada para um vetor da mesma vizinhança que o original :  $\delta_r = 0,8$

Para realizar uma comparação justa e avaliar como a inicialização dos vetores de peso afeta o desempenho dos algoritmos, dois métodos de inicialização de vetores de peso são empregados: o  $k$ -layers [Jiang and Yang, 2017] e o *Riesz S-Energy* [Blank et al., 2020], para todos os algoritmos. Os conjuntos iniciais de vetores de peso foram salvos como um arquivo externo, em que o  $k$ -layers foi implementado a partir do trabalho original e o *Riesz S-Energy* foi gerado a partir da plataforma Pymoo [Blank and Deb, 2020].

A *Inverted Generational Distance* (IGD) [Veldhuizen and Lamont, 1998] é a métrica utilizada para validar e comparar a qualidade das soluções geradas pelos algoritmos neste estudo. O IGD é uma métrica muito popular que avalia a convergência e a diversidade das soluções obtidas, sendo robusta a altas dimensões.

Cada algoritmo foi executado independentemente por 30 vezes para cada conjunto gerado pelos métodos de inicialização de vetores de peso e problema de teste. Para a comparação dos resultados, foi utilizado o teste estatístico não-paramétrico da soma de postos de Wilcoxon com um nível de significância de 5% em comparação com o MOEA/D-LNA. Os resultados são apresentados em tabelas divididas com cada grupo, em que as melhores médias são coloridas em azul e a última linha das tabelas demonstra o número de vitórias (+), perdas (-) e empates (=).

## 4.2 Configurações dos Problemas de Teste

Além de servirem para comparação dos algoritmos, os problemas de testes têm como principal objetivo verificar o comportamento do MOEA/D-LNA ao lidar com diferentes dimensões e em variadas fronteiras Pareto, principalmente as irregulares. Com isso, são interessantes problemas que apresentem diferenças da fronteira Pareto para o conjunto inicial de vetores de pesos, podendo assim ser observado a adaptação dos vetores ao longo das gerações. Além disso, busca-se verificar a capacidade dos algoritmos em manter a qualidade dos resultados em problemas onde não é necessário a adaptação dos vetores de pesos. Sendo assim, nos experimentos, são utilizados 32 problemas para validar e comparar o MOEA/D-LNA com os outros algoritmos da literatura: DTLZ1-7 [Deb et al., 2005], IDTLZ1-2 [Jain and Deb, 2014], WFG1-9 [Huband et al., 2006], as desenvolvidas para esta pesquisa, GPD1-2 (Apêndice A) e os problemas focados em muitos objetivos, MaF1-12 [Cheng et al., 2017]. Sendo este último conjunto de problemas de teste, MaF1-12, aplicado apenas nos algoritmos do grupo 2, que são aqueles com adaptação dos vetores de peso.

Estes conjuntos de problemas apresentam uma variedade interessante de características de fronteira Pareto: lineares, desconectadas, degeneradas, invertidas, etc; e todas as funções são escaláveis em relação ao espaço objetivo. O conjunto de problemas WFG, apresenta também, problemas de teste com domínios diferentes e características em que os pontos ótimos não estão localizados nas bordas do espaço de busca. Já os problemas de teste GPD, foram desenvolvidos com foco em fronteiras irregulares, adicionando desafios relacionados a mudança de posicionamento dos pontos extremos (GPD01) e similaridades com problemas desconectados (GPD02). Já os problemas MaF foram construídos buscando diversas propriedades de problemas reais e focados no conceito de muitos objetivos. As principais características dos problemas de teste são apresentadas na Tabela 1, bem como o número de variáveis de decisão ( $d$ ) em cada dimensão.

Os algoritmos foram testados em diferentes dimensões, 3, 5, 8 e 10, e como são utilizados dois métodos de geração de vetores de peso neste estudo, o  $k$ -layers e o Riesz S-Energy, os números de vetores de peso e consequentemente de indivíduos na população foram definidos de acordo com a Tabela 2. Para os problemas de teste DTLZ, GPD e MaF o número de funções de avaliação foram configurados como 60.000, 80.000, 100.000, e 120.000 para cada dimensão respectivamente. Já para os problemas de teste WFG, os números de funções de avaliação são configurados como 80.000, 100.000, 120.000 e 140.000, respectivamente.

Tabela 1 – Número de variáveis de decisão ( $d$ ) e características da fronteira Pareto para cada problema.

Problema	$d$	Fronteira	Bias
IDLTZ1	M - 1 + 5	Invertida	Não
IDLTZ2	M - 1 + 10	Invertida	Não
DTLZ1	M - 1 + 5	Linear	Não
DTLZ2	M - 1 + 10	Côncava	Não
DTLZ3	M - 1 + 10	Côncava	Não
DTLZ4	M - 1 + 10	Côncava	Sim
DTLZ5	M - 1 + 5	Degenerada	Sim
DTLZ6	M - 1 + 5	Degenerada	Sim
DTLZ7	M - 1 + 20	Desconectada	Não
WFG1	M - 1 + 10	Pontas Afiadas	Sim
WFG2	M - 1 + 10	Desconectada	Não
WFG3	M - 1 + 10	Desconectada	Não
WFG4	M - 1 + 10	Côncava	Não
WFG5	M - 1 + 10	Côncava	Não
WFG6	M - 1 + 10	Côncava	Não
WFG7	M - 1 + 10	Côncava	Sim
WFG8	M - 1 + 10	Côncava	Sim
WFG9	M - 1 + 10	Côncava	Sim
GPD01	M - 1 + 9	Dividida	Não
GPD02	M - 1 + 20	Gênero 1	Não
MaF1	M + 10 - 1	Invertida	Não
MaF2	M + 10 - 1	Côncava	Não
MaF3	M + 10 - 1	Côncava	Não
MaF4	M + 10 - 1	Invertida	Não
MaF5	M + 10 - 1	Convexa	Sim
MaF6	M + 10 - 1	Degenerada	Não
MaF7	M + 20 - 1	Desconectada	Não
MaF8	2	Degenerada	Não
MaF9	2	Degenerada	Não
MaF10	M + 10 - 1	Mista	Sim
MaF11	M + 10 - 1	Desconectada	Não
MaF12	M + 10 - 1	Côncava	Sim

### 4.3 Análise dos Parâmetros Adaptativos

O algoritmo MOEA/D-LNA possui alguns parâmetros que auxiliam no controle da adaptação dos vetores de peso durante o processo evolutivo. Os principais parâmetros que influenciam diretamente como a adaptação é realizada são o tamanho das vizinhanças locais e o ângulo de abertura de cada cone gerado pelo WVG,  $k$ ,  $\xi$ , respectivamente. Para investigar como esses parâmetros afetam o desempenho, foi comparado o IGD do MOEA/D-LNA com diferentes combinações de tamanhos da vizinhança local e do ângulo

Tabela 2 – Número de vetores de pesos e tamanho da população ( $N$ ) para cada método de inicialização.

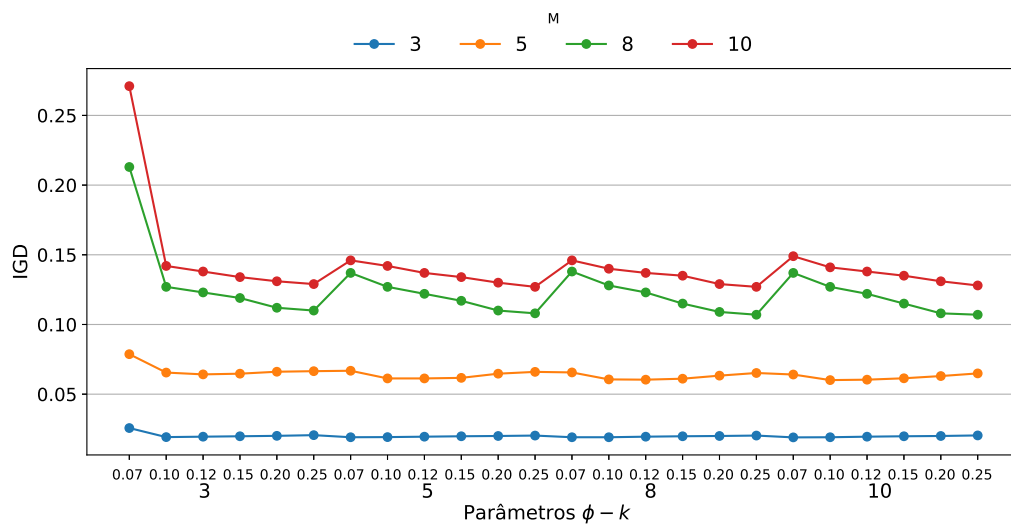
M	k-layers	Riesz S-Energy
3	106 ( $k = 7$ )	120
5	176 ( $k = 7$ )	160
8	217 ( $k = 6$ )	210
10	271 ( $k = 6$ )	300

de abertura. Como pretende-se redistribuir soluções de regiões populosas para outras mais promissoras, os valores investigados para  $k$  foram fixados em 3, 5, 8 e 10. Como  $\xi$  é a hiperdiagonal multiplicada pelo parâmetro  $\phi$ , aqui é também investigada a variação de  $\phi$  utilizando os valores de 0,07; 0,1; 0,15; 0,20 e 0,25. Para tanto, avaliamos esses parâmetros em 30 execuções em alguns problemas de teste irregulares: IDTLZ1, DTLZ5 e DTLZ7, sendo esses tipos um exemplo do objetivo principal do algoritmo.

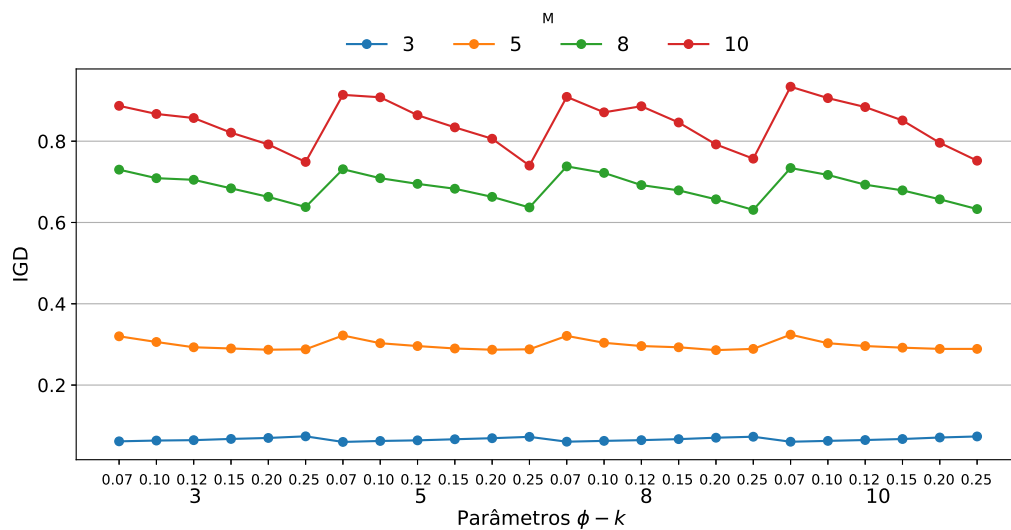
A Figura 6 apresenta a variação da média do IGD com os parâmetros  $k$  e  $\phi$  nos três problemas de teste. Os resultados mostram que para dimensões inferiores há pouca influência dos parâmetros no desempenho. Ao contrário, para dimensões maiores, um ângulo de abertura mais amplo indica melhora dos resultados, especialmente no problema de fronteira Pareto desconectada DTLZ7. Embora o valor da hiperdiagonal aumente com a dimensão do problema, observa-se que também é necessário um incremento de  $\phi$  de acordo com a dimensão. Assim, é proposto que  $\phi$  varia linearmente com o tamanho da dimensão  $M$ .

O tamanho da vizinhança local,  $k$ , parece não influenciar o desempenho nas funções IDTLZ1 e DTLZ7, mas na DTLZ5, um valor maior de  $k$  é recomendado conforme a dimensão aumenta. Portanto, sugere-se que  $k$  seja igual ao número da dimensão do problema,  $k = M$ , reduzindo também o número de parâmetros.

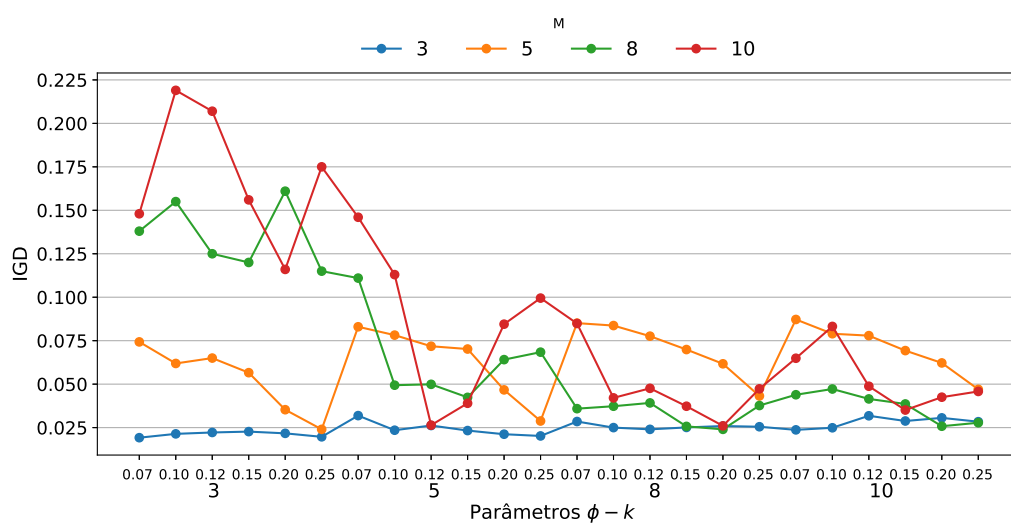
Seguindo a mesma ideia e utilizando o conhecimento adquirido do experimento anterior, outro parâmetro do MOEA/D-LNA que tem o objetivo de controlar a intensidade do procedimento de adaptação é a porcentagem de vetores de peso que podem ser alterados de uma só vez,  $\delta_c$ . Para analisar sua influência, quatro instâncias foram testadas com  $\delta_c$  igual a 0,2; 0,3; 0,4 e 0,5 nos problemas de teste irregulares anteriores. Na Figura 7 a variação da média dos problemas é apresentada e é perceptível que para a IDTLZ1, Figura 7a, e na DTLZ7, Figura 7b, há quase nenhuma alteração significativa. Porém, para a DTLZ5, Figura 7c, o aumento do número de vetores de peso máximo que podem mudar durante o processo de adaptação piora o valor do IGD, especialmente para dimensões maiores. Diante disso, é interessante que a adaptação dos vetores de peso ocorra gradativamente, assim recomenda-se que  $\delta_c$  varie entre 20% a 30%, permitindo que menos vetores sejam alterados por vez.



(a) IDTLZ1

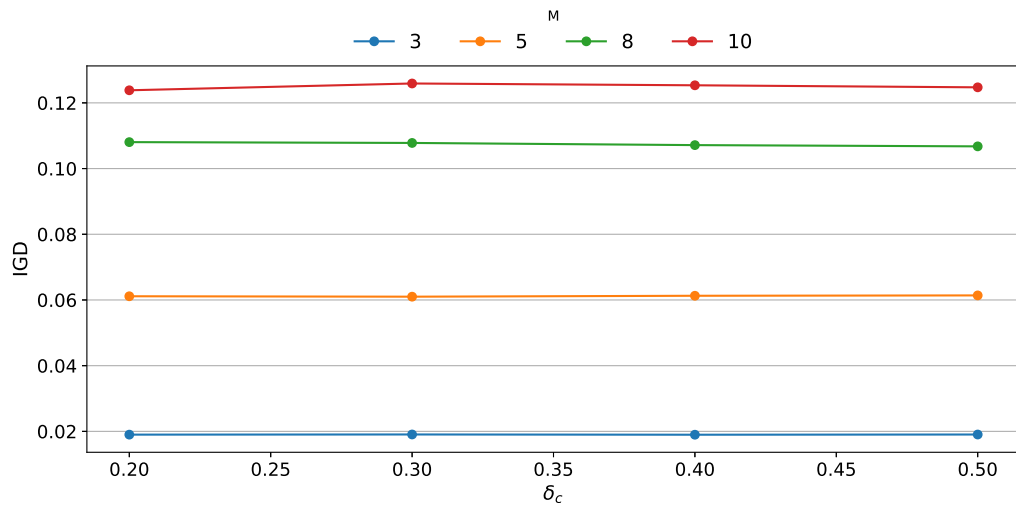


(b) DTLZ7

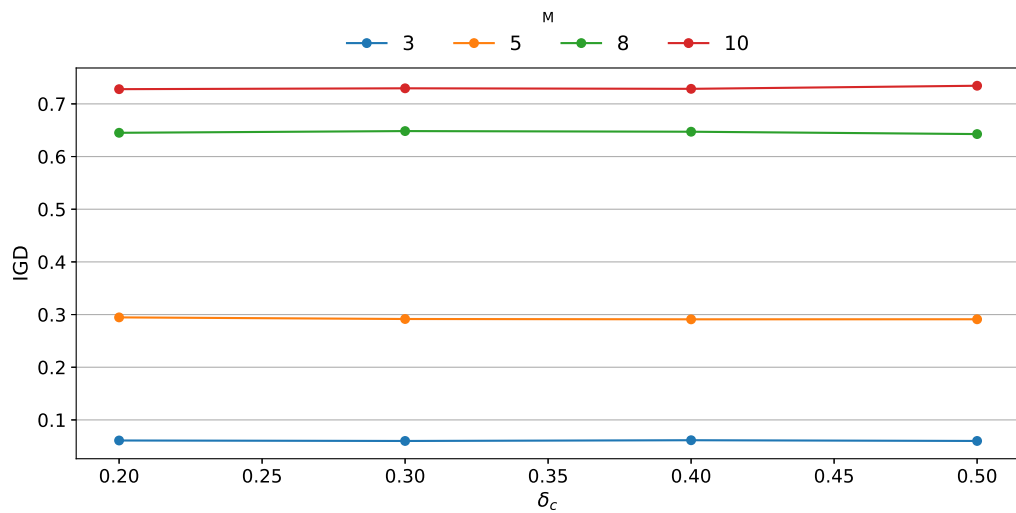


(c) DTLZ5

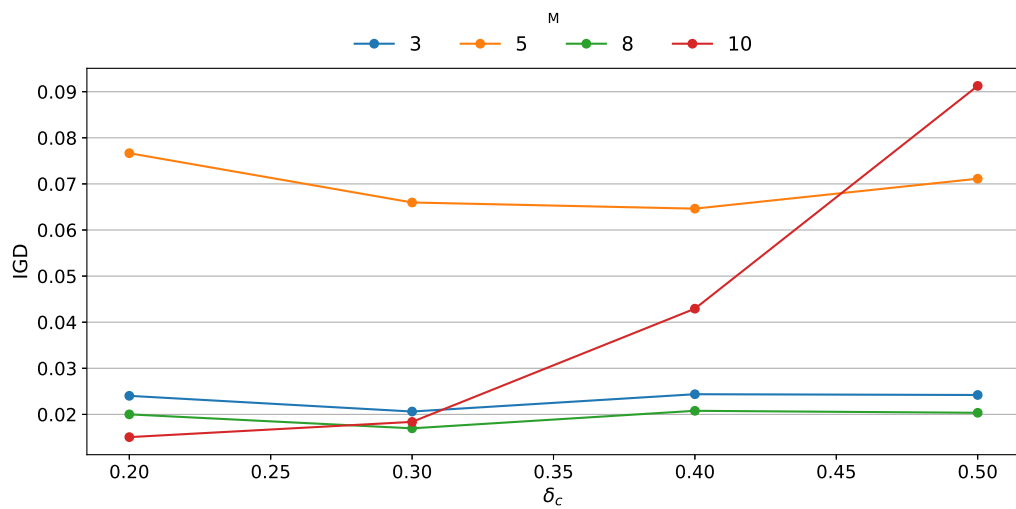
Figura 6 – Média do IGD para diferentes combinações dos parâmetros  $k$  e  $\phi$  do MOEA/D-LNA nos MOPs em diferentes dimensões (Fonte: Própria).



(a) IDTLZ1



(b) DTLZ7



(c) DTLZ5

Figura 7 – Média do IGD com diferentes valores de  $\delta_c$  do MOEA/D-LNA em MOPs e diferentes dimensões (Fonte: Própria).

No artigo original da métrica de melhoria [Camacho et al. \[2019\]](#), sugere-se usar o limite de 0,01 para  $M < 10$  e 0,001 para  $M \geq 10$  porém em testes, o segundo caso é considerado para todas as dimensões no MOEA/D-LNA (linha 22 no Algoritmo 2). Na Figura 8 um exemplo da variação da métrica de melhoria é apresentado no problema de teste DTLZ7 para 3 dimensões, em que os marcadores vermelhos indicam quando a atualização foi disparada. É perceptível que a evolução começa a estagnar em 20% a 25% do total da avaliação. Também como esperado, após cada atualização dos vetores de peso, há um aumento considerável no valor da métrica de melhoria, já que a população está se reorganizando e se adaptando ao novo conjunto de vetores de peso e algumas soluções acabam mudando de posição.

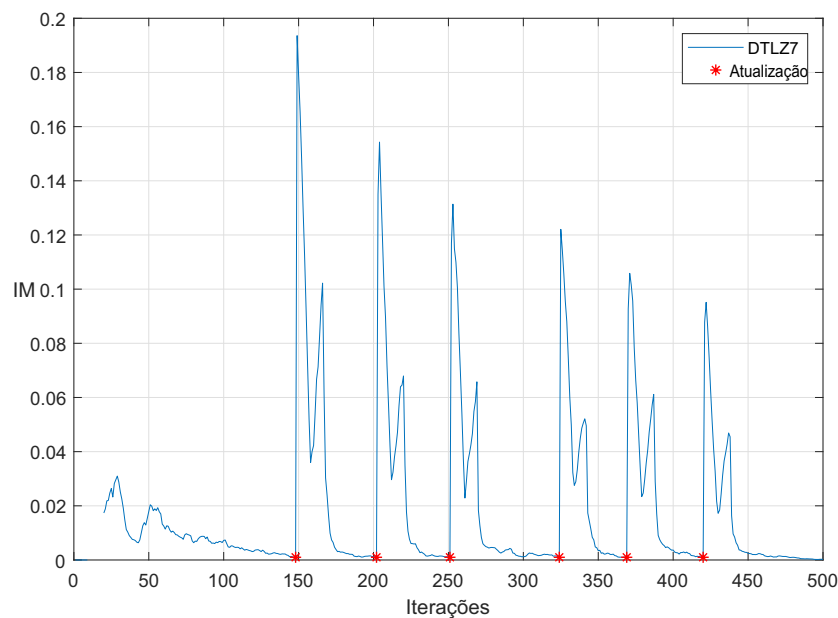


Figura 8 – Variação da métrica de melhoria (IM) no problema DTLZ7 em 3 dimensões (Fonte: Própria).

Na Figura 9 o mesmo padrão de variação é perceptível ao analisar o IGD durante a evolução, quando o MOEA/D-LNA aciona o processo de adaptação. Finalmente, a Figura 10 compara a distribuição dos vetores de peso do MOEA/D-LNA no início do processo de atualização e na etapa final, após todas as atualizações terem sido acionadas, para alguns problemas em 3 dimensões. Nestes exemplos, fica claro que as distribuições finais dos vetores de peso são semelhantes à forma da fronteira dos problemas que o algoritmo tenta aproximar.



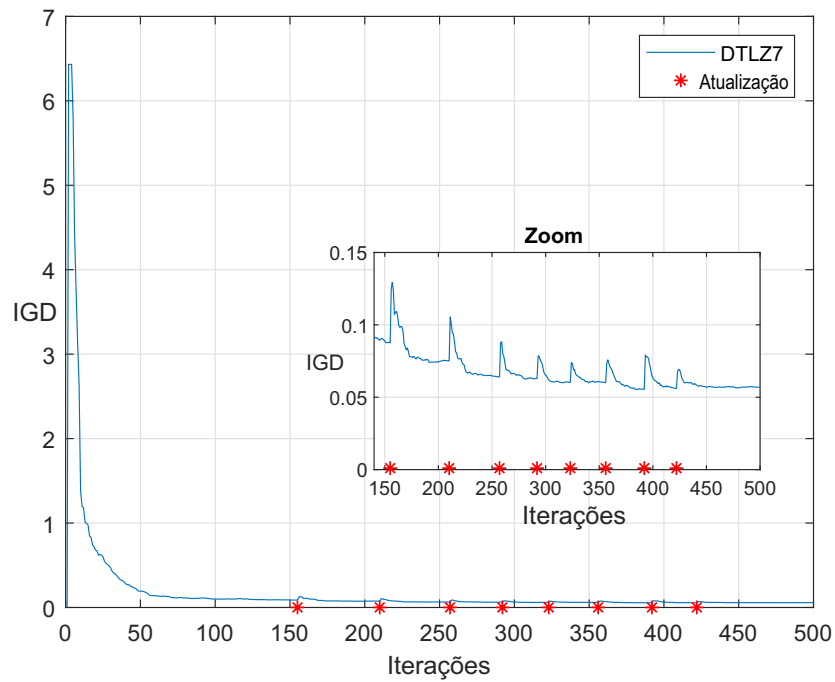


Figura 9 – Variação do IGD no problema DTLZ7 em 3 dimensões (Fonte: Própria).

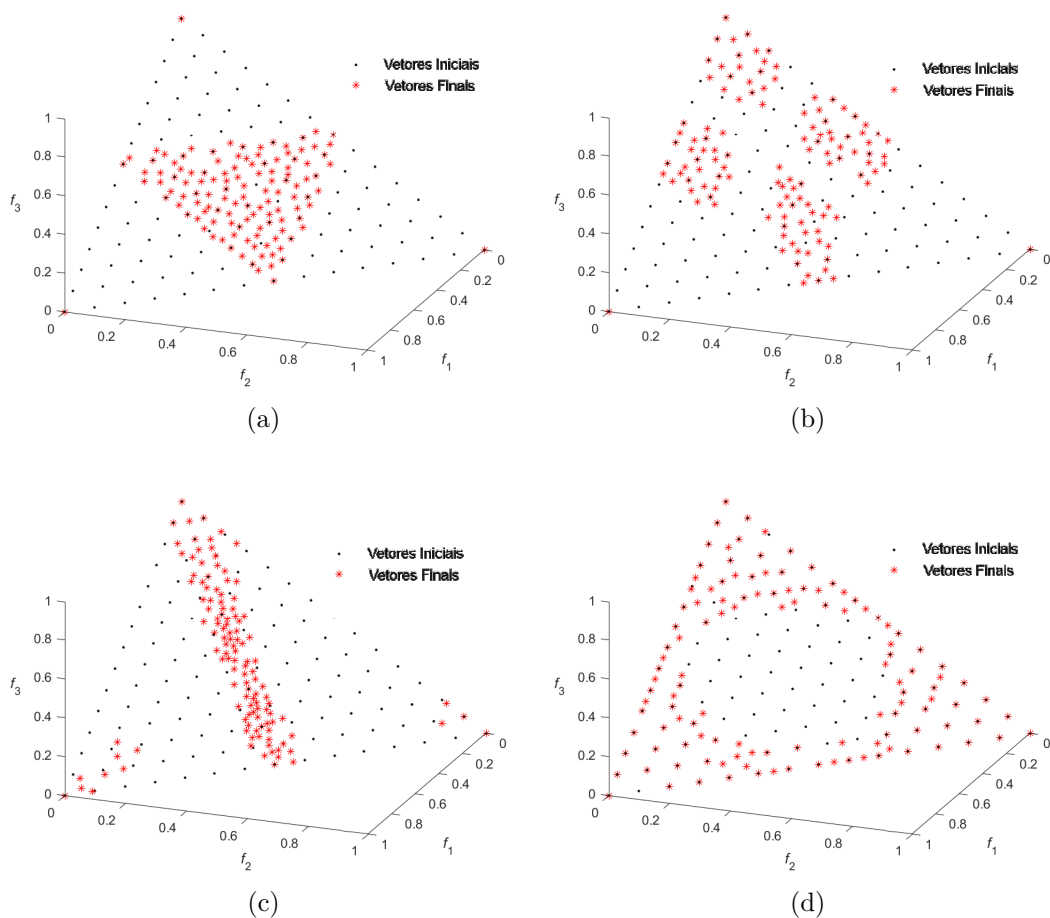


Figura 10 – Comparação entre as distribuições iniciais e finais dos vetores de peso do MOEA/D-LNA em diferentes problemas no espaço tridimensional: (a) IDTLZ1, (b) DTLZ7, (c) DTLZ5, (d) GPD02 (Fonte: Própria).

## 4.4 Resultados e Discussões

Nesta seção são apresentados os resultados obtidos dos experimentos com o MOEA/D-LNA e a comparação com outros algoritmos na resolução dos diferentes problemas de teste apresentados anteriormente. Os resultados e comparações, para ambos os métodos de inicialização de vetores de pesos, foram agrupados por classe de problemas e as principais discussões dos resultados são apresentadas na última seção. As tabelas de resultados do IGD de todos os experimentos são apresentadas no Apêndice B e o resumo de vitórias, perdas e empates é apresentado nas Tabelas 7 e 8.

### 4.4.1 Resultados nos Problemas DTLZ

Em uma primeira análise qualitativa, a Figura 11 apresenta as soluções encontradas pelo MOEA/D-LNA e outros dois MOEA nos problemas de teste DTLZ1, DTLZ5, DTLZ7 e IDTLZ1 em 3 dimensões usando o método de inicialização de vetores de pesos  $k$ -layers.

A média e o desvio padrão dos resultados da métrica IGD do MOEA/D-LNA e os outros dois grupos de MOEA com o método de inicialização  $k$ -layers e o método Riesz S-Energy nos problemas DTLZ são apresentados nas Tabela 9, Tabela 10 e Tabela 11, Tabela 12, respectivamente e os resumos das vitórias, perdas e empates nas Tabelas 7 e 8,

Analisando os resultados em comparação com os algoritmos do grupo 1, aqueles sem a estratégia de adaptação dos vetores de peso, com o método de inicialização de vetores,  $k$ -layers, presentes na Tabela 9, o MOEA/D-LNA alcançou 23 melhores médias em 36 instâncias de teste. Para o grupo 2, os métodos que possuem algum tipo de estratégia de adaptação de vetores de peso na Tabela 10, o MOEA/D-LNA alcançou 16 melhores médias de 36. O MOEA/D-LNA tem mais vitórias do que derrotas quando comparado aos concorrentes de todos os grupos. O melhor competidor, o MOEAD-AWA, venceu em 12 instâncias, seguido pelo MOEA/D com 10 vitórias. No problema de teste DTLZ1, o MOEAD/D-LNA demonstra melhora nos resultados apresentados pelo algoritmo base, MOEA/D, e está estatisticamente com resultados semelhantes ao MOEA/D-AWA e MOEADD na maioria das instâncias. Além disso, é notável que nas fronteiras irregulares dos IDTLZ1 e DTLZ7, o MOEA/D-LNA superou os concorrentes. Por outro lado, o desempenho no problema degenerado DTLZ5 só foi superior em 10 dimensões.

Comparando o MOEA/D-LNA com sua versão sem o procedimento de adaptação dos vetores de peso, MOEA/D-LNA\*, é possível perceber que a estratégia de adaptação é realmente eficaz e melhora os resultados do algoritmo. Mesmo o MOEA/D-LNA\* incorporando alguns aprimoramentos que ajudam em alguns casos na comparação com as versões base, MOEA/D ou MOEA/D-DE, eles não são suficientes para melhorar o desempenho

Tabela 3 – Número médio de atualizações ativadas para cada problema da família DTLZ no MOEA/D-LNA.

MOP	3	5	8	10
DTLZ1	14	7	1	1
DTLZ2	11	4	2	1
DTLZ3	1	1	1	1
DTLZ4	8	5	4	3
DTLZ5	3	4	14	14
DTLZ6	2	3	16	14
DTLZ7	7	4	2	2
IDTLZ1	6	7	10	9
IDTLZ2	7	5	6	5

geral, especialmente nos PFs irregulares.

Nos resultados usando o procedimento de inicialização do Riesz S-Energy, no primeiro grupo, Tabela 11, o MOEA/D-LNA obteve 14 melhores médias e no grupo 2, Tabela 12, obteve 16 melhores médias em 36 instâncias de teste. Embora haja algumas diferenças no número de vetores de peso entre os dois métodos, isso não é significativo, especialmente em dimensões superiores. Comparando com o grupo 1, o MOEA/D-LNA perdeu alguma das melhores médias em alguns casos, como no IDTLZ1 em 10 dimensões e nos DTLZ1 e DTLZ4. Em outros casos, o algoritmo obteve resultados muito melhores, como no DTLZ5 nas dimensões 8 e 10. A versão sem o procedimento de adaptação, MOEA/D-LNA\*, obteve desempenho muito melhor, e conquistou 4 melhores médias, sendo uma de fronteira Pareto irregular. Em uma análise geral, MOEA/D-LNA manteve mais vitórias do que derrotas contra os concorrentes. Para o grupo 2, a principal diferença está no aprimoramento da eficiência do algoritmo RVEA, que aumenta o número de vitórias em comparação ao MOEA/D-LNA. Além disso, é perceptível a piora dos resultados no caso do MOEA/D-AWA, o qual era o melhor competidor no outro experimento, e agora é o pior, mostrando que este algoritmo é claramente influenciado pela inicialização dos vetores de peso.

Finalmente, a Tabela 3 resume o número médio de vezes que o procedimento de atualização foi disparado durante a evolução do MOEA/D-LNA nos problemas DTLZ. É possível perceber que em dimensões menores o procedimento de atualização é acionado com mais frequência. Em alguns casos com fronteiras Pareto regulares, embora o algoritmo tenha acionado o processo de atualização muitas vezes, a distribuição final não foi afetada.

#### 4.4.2 Resultados nos Problemas WFG

Na Figura 12 os resultados qualitativos são exibidos para alguns problemas de teste da família WFG em que o MOEA/D-LNA é comparado a dois outros MOEA nos

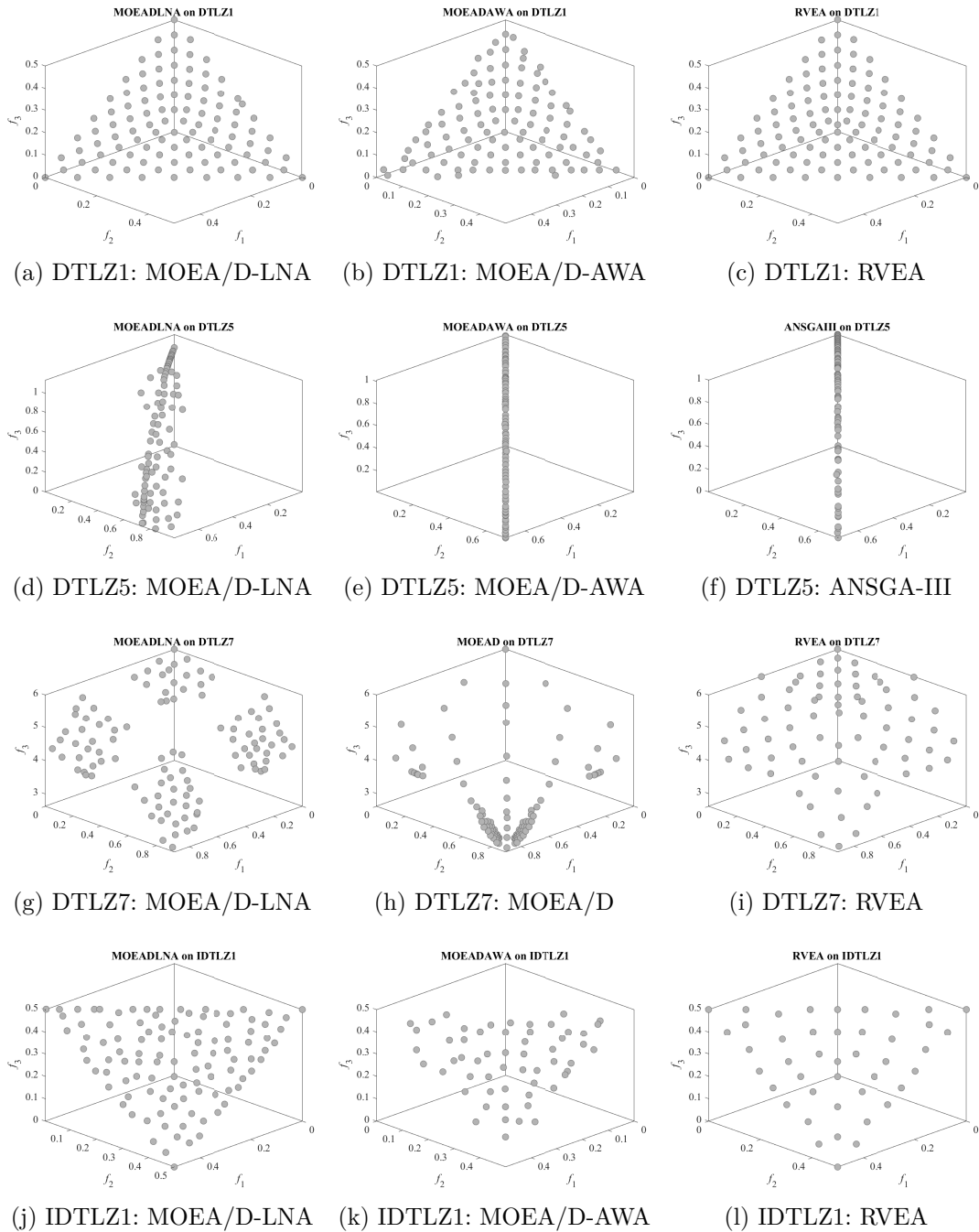


Figura 11 – Resultados com o melhor IGD para os problemas da DTLZ com o método de inicialização de vetores de peso  $k$ -layers (Fonte: Própria).

WFG3 e WFG2 em 3 dimensões e nos WFG9 e WFG5 em 10 dimensões através dos gráficos de coordenadas paralelas.

No problema WFG3, degenerado, é possível verificar dificuldades do MOEA/D-LNA e do ANSGA-III para aproximar a fronteira Pareto, em que algumas soluções não alcançaram a posição ótima no espaço de objetivos. Para o WFG9 em 10 dimensões, o MOEA/D-LNA aproxima de forma uniforme toda a fronteira Pareto com ótima cobertura, assim como o ANSGA-III e o RVEA. Uma análise interessante para o WFG2 é que o NSGA-III apresenta uma melhor aproximação do PF do que sua versão adaptativa, ANSGA-III e o MOEA/D-LNA.

Na sequência, as Tabelas 13 e 14 e as Tabelas 15 e 16 apresentam as médias e os desvios padrão dos resultados de IGD do MOEA/D-LNA e os outros dois grupos de MOEA com os métodos de inicialização *k*-layers e *Riesz S-Energy* nos problemas de teste WFG, respectivamente. Já os resumos de vitórias, perdas e empates são apresentados nas Tabelas 7 e 8.

Em comparação ao primeiro grupo, contendo aqueles MOEA sem adaptação de vetores de peso, o MOEA/D-LNA alcançou 15 melhores médias de 36 instâncias de teste. Analisando os resultados deste primeiro grupo, o MOEA/D-LNA venceu em praticamente todas as instâncias em que o NSGA-III perde, o melhor competidor. Contra os outros algoritmos, o MOEA/D-LNA teve mais vitórias do que derrotas. Comparando com os algoritmos do grupo 2, o MOEA/D-LNA ganhou 6 melhores médias de 36 e teve mais vitórias do que derrotas contra o ANSGA-III, o MOEAD-DRA e o EAG-MOEAD. O MOEA/D-AWA foi estatisticamente melhor do que MOEA/D-LNA, embora em muitos casos as diferenças sejam bem próximas.

Nos problemas de teste WFG, a versão sem adaptação dos vetores de peso, o MOEA/D-LNA\* teve alguns resultados interessantes, com 15 vitórias, 17 derrotas, 4 empates, e ainda apresentando 1 melhor média geral. Este resultado pode indicar que a adaptação dos vetores de peso não foi eficiente em algumas instâncias deste conjunto de teste.

Posteriormente, na análise dos resultados com o método de inicialização *Riesz S-Energy*, para o primeiro grupo, Tabela 15, o MOEA/D-LNA obteve 11 melhores médias em 36 instâncias. O melhor competidor ainda é o NSGA-III e o mesmo padrão da análise anterior também é observado aqui. Uma das principais surpresas foi o aumento no desempenho do MOEA/D-LNA\*, com 9 melhores médias de IGD. Ressalta-se que em alguns casos o desempenho de ambas as versões, com e sem procedimento de adaptação, é bastante semelhante. Nos resultados do grupo 2, Tabela 16, o MOEA/D-LNA melhorou o seu desempenho e apresentou 14 melhores médias de IGD. Os outros algoritmos também aumentaram o desempenho e, em muitos casos, os empates reverteram para vitórias para eles. A mesma redução de desempenho no MOEA/D-AWA é observada aqui, perdendo

muitas posições se comparado com os demais.

Conforme apresentado na primeira família de funções de teste, o número médio de atualizações acionadas nos problemas de teste WFG são apresentados na Tabela 4. Nestes resultados é possível perceber que em algumas instâncias do WFG1 e WFG2, o procedimento de atualização não foi acionado, a população não se estabilizou em nenhum momento. Esse resultado pode explicar o baixo desempenho do MOEA/D-LNA nesses problemas e resultados semelhantes se comparado ao MOEA/D-LNA\*. Outra característica interessante é que apenas uma ação da atualização pode não ser suficiente para aumentar o desempenho em alguns casos.

Tabela 4 – Número médio de atualizações ativadas para cada problema da família WFG no MOEA/D-LNA.

MOP	3	5	8	10
WFG1	0	1	2	0
WFG2	4	2	1	0
WFG3	4	5	3	2
WFG4	13	9	4	3
WFG5	16	6	4	3
WFG6	9	6	2	1
WFG7	6	3	3	2
WFG8	8	5	2	1
WFG9	13	6	4	4

### 4.4.3 Resultados nos Problemas GPD

Os resultados qualitativos para algumas instâncias dos problemas de teste GPD são apresentados na Figura 13 e na Figura 14. O MOEA/D-LNA é comparado a outros dois algoritmos em algumas instâncias, sendo apresentado o gráfico de coordenadas paralelas nas dimensões 10 e 5 das GPD01 e GPD02, respectivamente.

A ausência de informação de alguns pontos extremos no problema GPD01 atrapalha a evolução ou adaptação dos vetores de peso em alguns casos, como visto no exemplo do ANSGA-III. O MOEA/D-LNA não só é capaz de aproximar o formato da fronteira Pareto, mas também teve uma distribuição uniforme das soluções. O MOEA/D-AWA teve uma boa aproximação da fronteira do problema GPD01, mas também teve muitas soluções acumuladas em suas bordas. A mesma degradação pode ser observada no NSGA-III para o GPD01 em 10 dimensões. Ao contrário, o MOEA/D-LNA e o MOEA/D-AWA tiveram uma boa e uniforme cobertura da fronteira Pareto. Para o GPD02 em 3 dimensões, o MOEA/D-LNA encontrou um conjunto uniforme de soluções nas regiões ótimas da fronteira. O MOEA/D não conseguiu aproximar a fronteira neste problema. Por último, na instância de 5 dimensões da GPD02, o MOEA/D-LNA teve um resultado inferior ao

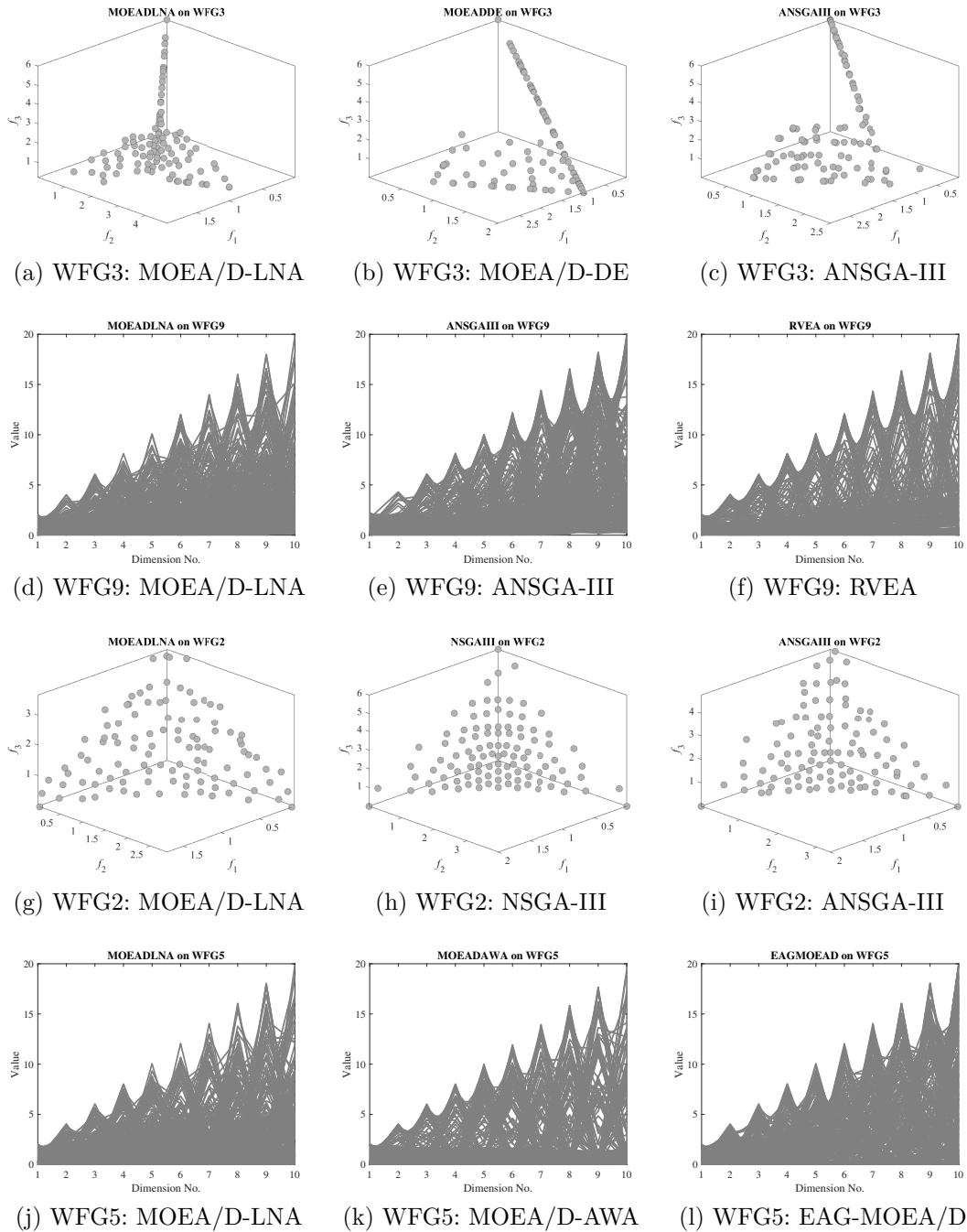


Figura 12 – Resultados com o melhor IGD para os problemas da WFG com o método de inicialização de vetores de peso  $k$ -layers (Fonte: Própria).

ANSGA-III e MOEA/D-AWA que teve uma distribuição realmente uniforme em toda a fronteira Pareto.

As Tabelas 17, 18 e Tabelas 19, 20 apresentam as médias e o desvios padrão dos resultados de IGD do MOEA/D-LNA e de outros dois grupos de MOEA com o  $k$ -layers e o *Riesz S-Energy* nos problemas de teste GPD, respectivamente. O resumo de vitórias, perdas e empates dos resultados são apresentados nas Tabelas 7 e 8,

Na comparação dos resultados com o método  $k$ -layers, Tabela 17 e 18, e o primeiro grupo, o MOEA/D-LNA alcançou 6 melhores médias em 8. O melhor competidor neste grupo foi o NSGA-III com 2 melhores médias mas apenas uma vitória sobre o MOEA/D-LNA. No segundo grupo, o MOEA/D-LNA obteve 7 melhores médias de 8. Assim, tendo mais vitórias do que derrotas na comparação com todos algoritmos entre os dois grupos, sendo de longe o melhor.

Para os resultados utilizando o *Riesz S-Energy*, Tabelas 19 e 20, o MOEA/D-LNA alcançou 3 melhores médias em ambos os grupos. O RVEA e o ANSGA-III obtiveram melhor desempenho com este método de inicialização. Além disso, conforme observado nos resultados anteriores nos problemas de teste WFG, o MOEA/D-LNA\*, versão sem procedimento de adaptação, teve um desempenho muito interessante através do IGD mas os resultados são visualmente piores, com má distribuição das soluções no espaço objetivo, como pode ser comparado na Figura 14a e Figura 14b.

Finalmente, os números médios de atualizações acionadas nos problemas de teste GPD são apresentados na Tabela 5. Nestes problemas de fronteira Pareto irregular, o procedimento de atualização do MOEA/D-LNA foi acionado pelo menos 2 vezes, e contribuiu para aumentar a diversidade das soluções na fronteira.

Tabela 5 – Número médio de atualizações ativadas para cada problema da GPD do MOEA/D-LNA.

MOP	3	5	8	10
GPD01	11	4	3	2
GPD02	6	3	2	2

#### 4.4.4 Resultados nos Problemas MaF

Nas Tabelas 21 e 22 são apresentados os resultados completos, média e desvio padrão, dos experimentos do MOEA/D-LNA e os outros MOEA no grupo 2 com os métodos de inicialização  $k$ -layers e o *Riesz S-Energy* nos problemas de teste MaF1-12, respectivamente. Como nos outros experimentos, o resumo de vitórias, perdas e empates dos resultados são apresentados na Tabela 8.



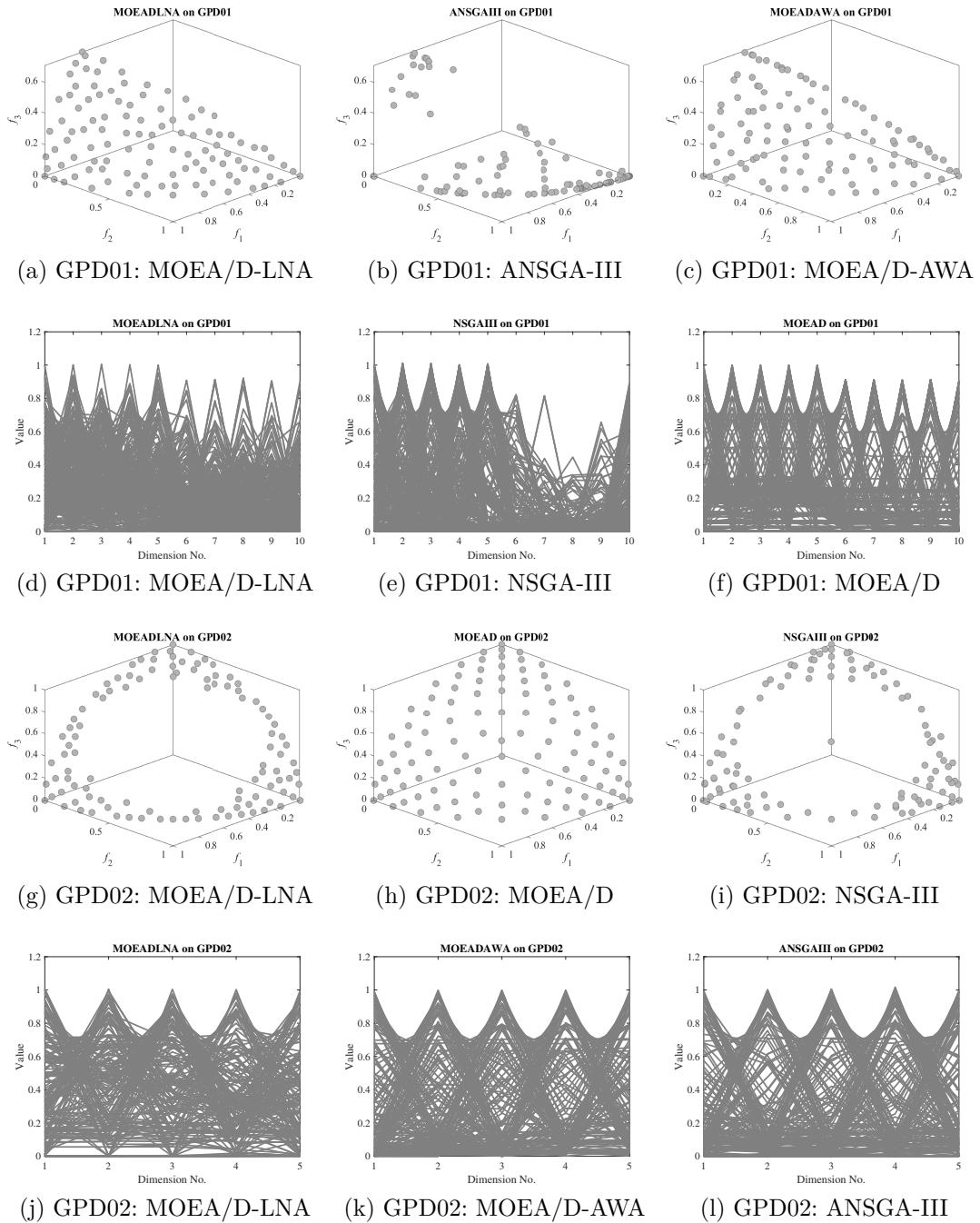


Figura 13 – Resultados com o melhor IGD para os problemas da GPD com o método de inicialização de vetores de peso  $k$ -layers (Fonte: Própria).

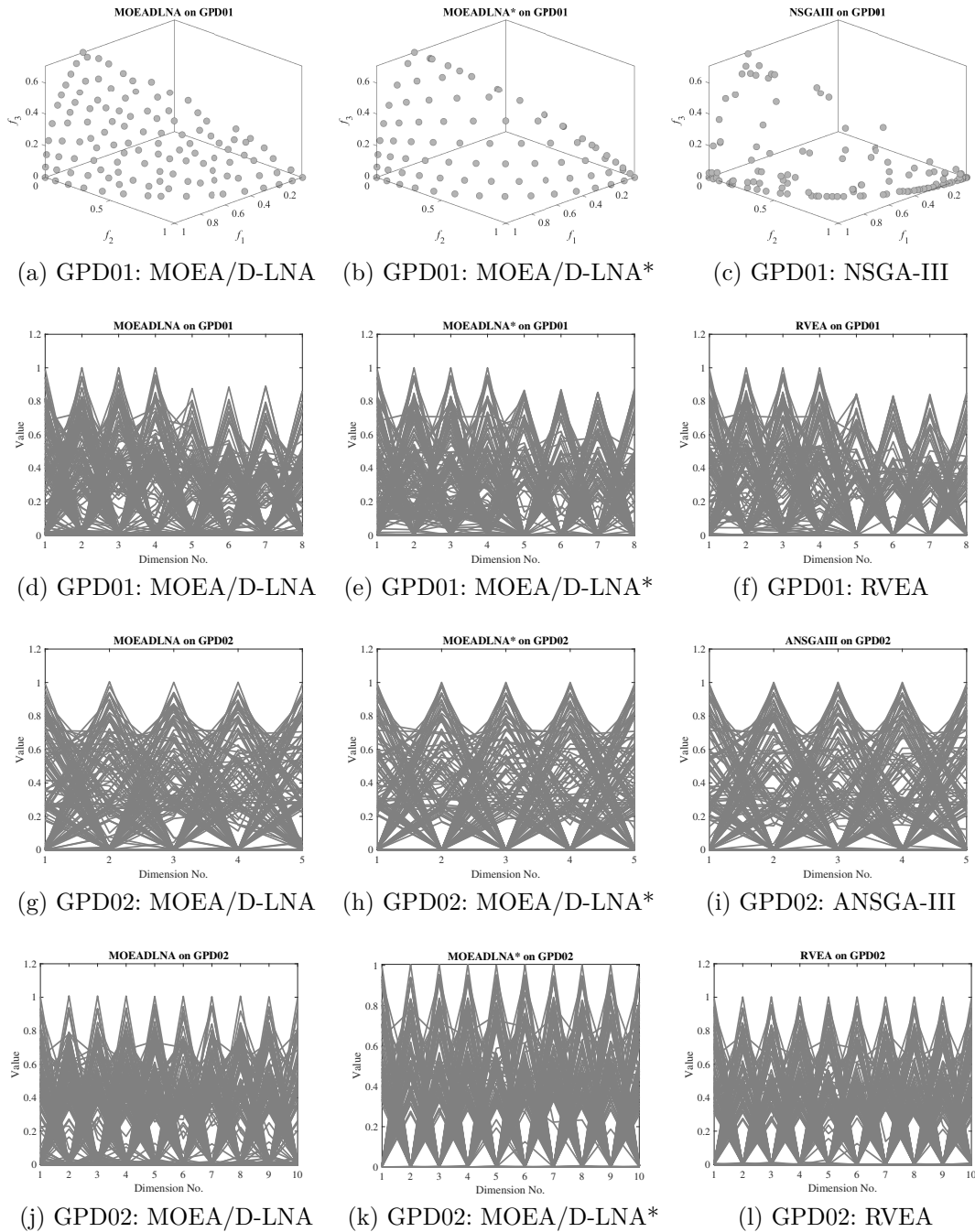


Figura 14 – Resultados com o melhor IGD para os problemas da GPD com o método de inicialização de vetores de peso Riesz S-Energy (Fonte: Própria).

No primeiro conjunto de resultados apresentados na Tabela 21, com o  $k$ -layers, o MOEA/D-LNA alcançou 12 melhores médias em 48 instâncias de teste. Nessa linha, o melhor competidor foi o ANSGA-III com 9 melhores médias e o pior foi o RVEA com 4 melhores médias. O MOEA/D-LNA tem mais vitórias do que derrotas quando comparado a todos os outros competidores individualmente. Em algumas instâncias de teste em que MOEA/D-LNA não foi o melhor, ele foi o segundo melhor, como no MaF4.

Na sequência, a análise dos resultados adotando o *Riesz S-Energy*, Tabela 22 indicou que o MOEA/D-LNA alcançou 19 melhores médias em 48 instâncias de teste. A adição de 6 vitórias ocorreu principalmente nos testes MaF4, MaF6 e MaF12, mas algumas outras instâncias foram perdidas ao comparar os resultados com o outro método de inicialização. O RVEA foi o melhor competidor com 11 melhores médias, seguido pela ANSGA-III. Com este método de inicialização também o MOEA/D-LNA teve mais vitórias do que derrotas contra todos os outros concorrentes. Além disso, neste conjunto de testes é observada a mesma degradação dos resultados encontrados nos demais experimentos com MOEA/D-AWA, sendo o pior algoritmo.

Finalmente, o número médio de atualizações ativadas nos problemas de teste MaF é apresentado na Tabela 6. Na maioria dos casos, o procedimento de atualização foi acionado pelo menos uma vez. Além disso, mais disparos foram executados nos problemas MaF6, MaF8 e MaF9, todos com fronteira Pareto degenerada. Outro ponto é que a dimensão parece influenciar também no número de atualizações; portanto, os problemas em dimensões superiores são atualizados menos do que em dimensões inferiores em muitos casos deste conjunto de teste.

Tabela 6 – Número médio de atualizações ativadas para cada problema da família MaF no MOEA/D-LNA.

MOP	3	5	8	10
MaF1	9	7	9	11
MaF2	7	4	4	4
MaF3	1	1	1	2
MaF4	2	2	2	2
MaF5	10	2	4	3
MaF6	8	11	14	14
MaF7	9	4	3	3
MaF8	4	6	9	10
MaF9	5	5	6	7
MaF10	0	1	1	1
MaF11	3	1	2	2
MaF12	8	4	3	3

### 4.4.5 Discussões dos Resultados

Nas seções anteriores foi possível verificar que o MOEA/D-LNA teve bom desempenho em muitos conjuntos de teste distintos e fronteiras Pareto com propriedades diferentes. Portanto, nesta seção são discutidos alguns dos resultados mais interessantes encontrados.

No problema DLTZ1, fronteira regular e linear, é possível verificar que embora o MOEA/D-LNA tenha uma estratégia de adaptação de vetores de peso, dificilmente esta afeta a distribuição das soluções na fronteira Pareto como foi observado no caso do MOEA/D-AWA. Isso ocorre porque quando o processo de adaptação é acionado, após  $t_{on}$  gerações e a métrica de estagnação da população estabiliza, a maioria dos vetores de peso tem apenas uma solução em sua vizinhança local. Conseqüentemente, a distribuição dos vetores de peso dificilmente muda.

Embora em dimensões mais altas de alguns problemas de teste como no DTLZ5 ou no MaF6, o MOEA/D-LNA tenha obtido os melhores resultados de IGD, em dimensões mais baixas, por outro lado, obteve resultados ruins. Além disso, é possível notar que as soluções finais não estão todas localizadas na fronteira Pareto, apresentando desempenho visualmente pior que o MOEA/D-AWA e o ANSGA-III como pode ser visto na Figura 11. Os resultados em outras fronteiras degeneradas, como os problemas DTLZ6, WFG3, MaF8 e MaF9 também foram abaixo dos competidores. Uma possível explicação para este comportamento, é que no processo de adaptação do MOEA/D-LNA, são adaptados todos os vetores de peso com vizinhanças locais com tamanho maior que um; portanto, gerando um cone de vetores de peso para cada um e espalhando os vetores de peso ao redor da fronteira Pareto degenerada, deixando algumas partes da fronteira sem nenhum vetor de peso. Além disso, conforme descrito na Seção 4.3 da análise dos parâmetros, a fronteira degenerada aparenta ser muito sensível ao ângulo do cone de vetores de peso gerado durante o processo de adaptação. Portanto, o tamanho do ângulo fixo do cone pode influenciar o método ao tentar cobrir este tipo de fronteira Pareto.

Para as fronteiras Pareto desconectadas como a DLTZ7 e a MaF7, é perceptível uma melhor distribuição das soluções no espaço objetivo e nos resultados IGD encontrados pelo MOEA/D-LNA. Nestes casos, a adaptação dos vetores de peso redistribuiu as soluções que tendem a se acumular nas bordas de algumas regiões como visto nos resultados do MOEA/D. A responsável por este resultado é a limitação do tamanho das vizinhanças locais, pois as soluções excedentes são realocadas para outras vizinhanças locais de menor tamanho, contribuindo com a geração de novos vetores de peso em regiões pouco exploradas. Uma característica semelhante é observada nos problemas IDTLZ1, MaF1, GPD1, GPD2 e MaF4, permitindo que um número mais diversificado de soluções sejam distribuídas ao longo da fronteira Pareto. Ainda, nos resultados do RVEA ou MOEA/D-AWA, por exemplo, podem ser observadas muitas soluções semelhantes ou até mesmo iguais,

reduzindo expressivamente a diversidade dos resultados finais. No MOEA/D-LNA este problema é minimizado pois duas ou mais soluções realmente próximas farão parte da mesma vizinhança local e conseqüentemente gerando um novo cone de vetores de peso na atualização seguinte, aumentando a diversidade de soluções finais.

Analisando o número de atualizações acionadas é possível perceber que em dimensões maiores o número de atualizações é reduzido nos diversos problemas de teste. Este comportamento provavelmente se deve ao aumento do tamanho do espaço de busca em dimensões superiores, pois as populações demoram mais gerações para estagnar, influenciando as condições para iniciar as atualizações no MOEA/D-LNA conforme discutido na seção 3.4. Nestes resultados é possível perceber também que em algumas instâncias do WFG1 e WFG2, o procedimento de atualização não foi acionado, indicando que a população não se estabilizou em nenhum momento. Em outros, como o DTLZ3, MaF3 e o MaF10, a atualização foi acionada apenas algumas vezes, 1 a 3 vezes. Isso pode explicar o baixo desempenho nesses problemas. Outro resultado interessante é que apenas uma atualização pode não ser suficiente para melhorar os resultados em alguns casos e pode até deteriorar em outros.

Ao comparar o MOEA/D-LNA com sua versão sem o procedimento de atualização de vetores de peso, denominado de MOEA/D-LNA\*, é perceptível a melhora geral em muitos resultados. Em alguns casos menos comuns, como no GPD01, DTLZ1 ou DTLZ5 com *Riesz S-Energy*, o MOEA/D-LNA\* alcançou resultados muito próximos ao MOEA/D-LNA, às vezes sendo estatisticamente melhor. No entanto, foi possível observar que apenas os resultados do IGD foram semelhantes, mas a distribuição das soluções foi muito melhor na versão adaptativa. Em resumo, os outros procedimentos implementados a partir da literatura ajudaram, mas o procedimento de adaptação fez diferença no aprimoramento dos resultados.

Por fim, no aspecto dos métodos de inicialização, embora o MOEA/D-LNA tenha perdido algumas posições para outros concorrentes em alguns problemas e ganhado em outros ao alterar o método, entende-se que os resultados gerais indicam que o processo de adaptação ajuda a evolução. Outro ponto interessante observado, é que o MOEA/D-LNA foi realmente robusto na hora em que o método de inicialização é alterado. O MOEA/D-AWA, por exemplo, foi completamente influenciado pela mudança de seu método de inicialização e teve um desempenho ruim com o *Riesz S-Energy*. Por causa dos resultados mais consistentes em muitos algoritmos e no MOEA/D-LNA, a recomendação é o método *k-layers*.

Tabela 7 – Resumo das vitórias (+), perdas (-) e empates (=) dos resultados de IGD de cada algoritmo do grupo 1 em comparação ao MOEA/D-LNA utilizando o teste estatístico não-paramétrico da soma dos postos de Wilcoxon com um nível de significância de 5%.

Método	Problema	MOEAD	MOEADDE	MOEADM2M	MOEADD	NSGAIII	MOEADLNA*
<i>k-layers</i>	DTLZ	10/25/1	8/27/1	1/35/0	4/28/4	8/25/3	8/24/4
	WFG	5/27/4	4/31/1	4/30/2	10/24/2	18/15/3	15/17/4
	GPD	0/8/0	0/8/0	0/8/0	0/7/1	1/6/1	0/8/0
Riesz S-Energy	DTLZ	10/23/3	9/26/1	1/35/0	11/19/6	12/20/4	11/16/9
	WFG	7/27/2	2/34/0	5/30/1	13/20/3	21/13/2	15/10/11
	GPD	3/4/1	0/8/0	0/8/0	3/4/1	2/6/0	5/3/0

Tabela 8 – Resumo das vitórias (+), perdas (-) e empates (=) dos resultados de IGD de cada algoritmo do grupo 2 em comparação ao MOEA/D-LNA utilizando o teste estatístico não-paramétrico da soma dos postos de Wilcoxon com um nível de significância de 5%.

Método	Problema	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA
<i>k-layers</i>	DTLZ	12/20/4	9/26/1	4/29/3	5/27/4	4/30/2
	WFG	21/7/8	4/29/3	6/25/5	13/16/7	15/19/2
	GPD	1/6/1	0/8/0	0/8/0	1/6/1	0/8/0
	MaF	15/27/6	11/31/6	16/28/4	17/27/4	12/33/3
Riesz S-Energy	DTLZ	2/33/1	9/27/0	5/28/3	12/23/1	11/23/2
	WFG	7/25/4	2/34/0	7/27/2	6/16/4	18/15/3
	GPD	0/8/0	0/8/0	0/8/0	3/5/0	4/3/1
	MaF	2/43/3	7/40/1	11/28/9	18/23/7	14/29/5

## 4.5 Considerações Parciais

Este Capítulo apresentou as análises com relação aos parâmetros específicos do MOEA/D-LNA relacionados ao procedimento de adaptação de vetores de pesos. Apresentou-se também os resultados dos testes realizados para validação e comparação do algoritmo proposto com outros algoritmos estado da arte. Os competidores foram divididos em dois grupos, com e sem procedimento de atualização de vetores de pesos. Os pontos principais e contrastes foram destacados e discutidos.

No Capítulo seguinte, é realizado o fechamento destes estudos com um resumo dos temas abordados e as principais conclusões extraídas. Também são apresentadas sugestões de trabalhos e estudos futuros.

# Capítulo 5

## Conclusões

Os algoritmos genéticos para problemas multi-objetivo vem sendo cada vez mais pesquisados e implementados para a resolução de diversos problemas de engenharia ao longo do tempo. Em específico, os algoritmos baseados em decomposição e funções de agregação demonstram em estudos características promissoras, principalmente, para a resolução de problemas em dimensões elevadas quando comparados aos algoritmos baseados em dominância Pareto. Muito desta popularização se deve ao algoritmo proposto por [Zhang and Li \[2007\]](#), o MOEA/D, que aplicou os conceitos de decomposição em uma estrutura simples e eficiente.

Na continuidade destes estudos, verificou-se que um dos principais problemas da abordagem baseada em decomposição é a necessidade de uma boa distribuição inicial do conjunto de vetores de pesos. Quanto mais próximo esse conjunto é da fronteira Pareto melhores serão os resultados obtidos pelo algoritmo, fazendo com que o algoritmo dependa fortemente do tipo da fronteira Pareto apresentada pelo problema. Contudo, geralmente a fronteira Pareto não é conhecida e em problemas reais tendem a ser do tipo irregulares (desconectadas, degeneradas e invertidas), fazendo com que o conjunto inicial de vetores de peso não contemple partes da fronteira Pareto. Uma das metodologias apresentadas na literatura para contornar estes problemas é a adaptação do conjunto de vetores de pesos em conjunto com a população.

Com isso, este trabalho propôs implementar um algoritmo, denominado de MOEA/D-LNA, que alia a estrutura apresentada no clássico MOEA/D com uma mecânica de atualização de vetores de pesos. Para esta realização, foram inicialmente estudados os trabalhos que envolvem o contexto de algoritmos baseados em decomposição e agregação e aqueles que utilizam técnicas para adaptação dos vetores de peso. Foram também investigadas técnicas que pudessem auxiliar na adaptação dos vetores, como metodologias de inicialização de vetores de pesos, normalizações, métricas de estagnação e dentre outras, apresentados no Capítulo 2.

No Capítulo 3 é detalhada a construção do algoritmo MOEA/D-LNA, assim como



todas as técnicas utilizadas para auxiliar na etapa de adaptação de vetores de pesos. Em resumo, o algoritmo trabalhou com a relação de proximidade das soluções não-dominadas armazenadas em um arquivo externo com os vetores de peso, construindo o conceito denominado de vizinhanças locais. Neste aspecto, o procedimento de adaptação é dividido em quatro etapas principais: 1) identificar as regiões promissoras e construir as vizinhanças locais, 2) remover os vetores não promissores, 3) gerar um novo conjunto de vetores de pesos e 4) selecionar os melhores vetores e associar a população corrente.

Na conclusão deste estudo, a Seção 5.1 apresenta os principais resultados, positivos e negativos, encontrados durante as comparações experimentais do Capítulo 4. Na Seção 5.2 são sugeridos pontos de pesquisa e melhoria e, finalmente, na Seção 5.3, são apresentados as publicações provenientes desta dissertação.

## 5.1 Principais Resultados

O MOEA/D-LNA foi comparado a outros algoritmos, tanto com e sem atualização de vetores de pesos, em 148 instâncias de problemas de teste divididos em 32 problemas em diferentes dimensões e utilizando 2 métodos de inicialização de vetores de pesos diferentes. Estes experimentos foram analisados tanto sob a ótica quantitativa, através da métrica IGD, quando de análises qualitativas, através da distribuição de soluções e ajuste dos vetores para o MOEA/D-LNA.

Os resultados experimentais, Capítulo 4, demonstraram que o algoritmo foi capaz de identificar e ajustar os vetores de peso conforme o formato da fronteira Pareto. Além disso, os resultados mostraram melhor desempenho nas fronteiras irregulares, principalmente nas desconectadas e invertidas. Em comparação aos outros algoritmos, o MOEA/D-LNA se mostrou competitivo e em alguns casos até superior.

As comparações entre dois diferentes métodos de inicialização de vetores de pesos possibilitou verificar a inconsistência de alguns algoritmos quando os pesos são alterados, resultando em deterioração dos resultados. Neste aspecto, o MOEA/D-LNA se mostrou robusto, apresentando resultados proporcionais em cada caso.

Apesar das melhorias apresentadas, o MOEA/D-LNA enfrentou dificuldades nos problemas irregulares degenerados. Os resultados foram inferiores em muitos casos, mesmo apresentando uma distribuição de vetores de pesos nas regiões onde se encontrava a fronteira Pareto estimada. Outro ponto, é o número baixo de atualizações ativadas em certos problemas, como nas WFG1 e WFG2, que também resultaram em piores resultados para o MOEA/D-LNA, indicando problema não com a adaptação mas possivelmente com a métrica de estagnação.

## 5.2 Futuras Investigações

Dada a proposta deste trabalho e tendo em vista a extensão e melhoria da área de algoritmos genéticos multi-objetivo no contexto de algoritmos baseados em decomposição, são sugeridos alguns tópicos de estudo futuro:

1. Investigar o desempenho do MOEA/D-LNA em outros problemas de teste e também em aplicações de mundo real.
2. Investigar o uso de uma inicialização de vetores de pesos totalmente aleatória e como a adaptação dos vetores de pesos influencia neste tipo de inicialização.
3. Desenvolver uma variação na etapa 3 do procedimento de adaptação dos vetores de peso, onde o ângulo de abertura do cone WVG seria ajustado conforme o tamanho da vizinhança local, e não somente pelo aumento da dimensão.
4. Investigar o custo computacional empregado nas operações de adaptação dos vetores de peso. Principalmente, no que diz respeito à geração dos cones WVG.
5. Investigar uma possível nova abordagem em que é feita a reinicialização do MOEA/D-LNA, substituindo o conjunto de vetores de pesos inicial pelo conjunto que foi atualizado até um dado momento da otimização. Fazendo assim, uma versão de duas fases, uma para achar um conjunto de vetores de pesos aproximado e a outra para otimizar as soluções.
6. Investigar melhor as razões das dificuldades encontradas em problemas degenerados mesmo quando ajustando os vetores para as regiões de interesse.
7. Investigar a utilização de outros parâmetros, como por exemplo, remover o limite do tamanho do arquivo externo.

## 5.3 Publicações

Os seguintes trabalhos científicos foram aceitos para publicação durante a elaboração desta dissertação:

- **Junqueira, P. P.**, Meneghini, I. R. and Guimarães, F. G., Local Neighborhood-Based Adaptation of Weights in Multi-Objective Evolutionary Algorithms Based on Decomposition. *2021 IEEE Congress on Evolutionary Computation* (pp. 1454-1461), (CEC), 2021.

- **Junqueira, P. P**, Meneghini, I. R. and Guimarães, F. G., Multi-Objective Evolutionary Algorithm based on Decomposition with an External Archive and Local-Neighborhood Based Adaptation of Weights. Swarm and Evolutionary Computation (em revisão)

# Referências

- R. Agrawal, K. Deb, and R. Agrawal. Simulated binary crossover for continuous search space. *Complex Systems*, 9, 06 2000.
- E. Alba, B. Dorronsoro, F. Luna, A. Nebro, P. Bouvry, and L. Hogie. A cellular multi-objective genetic algorithm for optimal broadcasting strategy in metropolitan manets. *Computer Communications*, 30, 02 2007. doi: 10.1109/IPDPS.2005.4.
- M. Asafuddoula, H. K. Singh, and T. Ray. An enhanced decomposition-based evolutionary algorithm with adaptive reference vectors. *IEEE Transactions on Cybernetics*, 48(8): 2321–2334, 2018. doi: 10.1109/TCYB.2017.2737519.
- J. Bader and E. Zitzler. Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary computation*, 19:45–76, 03 2011. doi: 10.1162/EVCO\_a\_00009.
- J. Blank and K. Deb. Pymoo: Multi-objective optimization in python. *IEEE Access*, 8: 89497–89509, 2020. doi: 10.1109/access.2020.2990567.
- J. Blank, K. Deb, and P. C. Roy. Investigating the normalization procedure of NSGA-III. In *Evolutionary Multi-Criterion Optimization - 10th International Conference, EMO 2019, East Lansing, MI, USA, March 10-13, 2019, Proceedings*, pages 229–240, 2019. doi: 10.1007/978-3-030-12598-1\_19. URL [https://doi.org/10.1007/978-3-030-12598-1\\_19](https://doi.org/10.1007/978-3-030-12598-1_19).
- J. Blank, K. Deb, Y. Dhebar, S. Bandaru, and H. Seada. Generating well-spaced points on a unit simplex for evolutionary many-objective optimization. *IEEE Transactions on Evolutionary Computation*, PP:1–1, 01 2020. doi: 10.1109/TEVC.2020.2992387.
- X. Cai, Y. Li, Z. Fan, and Q. Zhang. An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization. *IEEE Transactions on Evolutionary Computation*, 19(4):508–523, aug 2015. doi: 10.1109/tevc.2014.2350995.

- X. Cai, Z. Mei, and Z. Fan. A decomposition-based many-objective evolutionary algorithm with two types of adjustments for direction vectors. *IEEE Transactions on Cybernetics*, 48(8):2335–2348, 2018. doi: 10.1109/TCYB.2017.2737554.
- A. Camacho, G. Toscano, R. Landa, and H. Ishibuchi. Indicator-based weight adaptation for solving many-objective optimization problems. In *Lecture Notes in Computer Science*, pages 216–228. Springer International Publishing, 2019. doi: 10.1007/978-3-030-12598-1\_18.
- R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff. A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 20, 01 2016. doi: 10.1109/TEVC.2016.2519378.
- R. Cheng, M. Li, Y. Tian, X. Zhang, Y. Jin, and X. Yao. A benchmark test suite for evolutionary many-objective optimization. *Complex and Intelligent Systems*, 3:67–81, 03 2017. doi: 10.1007/s40747-017-0039-7.
- C. Coello. A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems*, page 269308, 1999. doi: 10.1007/BF03325101.
- I. Das and J. Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8, 07 2000. doi: 10.1137/S1052623496307510.
- L. R. de Farias and A. F. Araújo. A decomposition-based many-objective evolutionary algorithm updating weights when required. *Swarm and Evolutionary Computation*, page 100980, 2021. ISSN 2210-6502. doi: <https://doi.org/10.1016/j.swevo.2021.100980>. URL <https://www.sciencedirect.com/science/article/pii/S2210650221001425>.
- L. R. C. de Farias, P. H. M. Braga, H. F. Bassani, and A. F. R. Araújo. Moea/d with uniformly randomly adaptive weights. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '18*, page 641648, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356183. doi: 10.1145/3205455.3205648. URL <https://doi.org/10.1145/3205455.3205648>.
- K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2): 182–197, April 2002. ISSN 1941-0026. doi: 10.1109/4235.996017.
- K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable test problems for evolutionary multiobjective optimization. In *Advanced Information and Knowledge Processing*, pages 105–145. Springer-Verlag, 2005. doi: 10.1007/1-84628-137-7\_6.

- K. Deb, K. Miettinen, and S. Chaudhuri. Toward an estimation of nadir objective vector using a hybrid of evolutionary and local search approaches. *IEEE Transactions on Evolutionary Computation*, 14(6):821–841, 2010. doi: 10.1109/TEVC.2010.2041667.
- K. Deb, Q. Zhang, and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *Evolutionary Computation, IEEE Transactions on*, 18:577–601, 08 2014. doi: 10.1109/TEVC.2013.2281535.
- K. Deb, S. Bandaru, and H. Seada. Generating uniformly distributed points on a unit simplex for evolutionary many-objective optimization: 10th international conference, emo 2019, east lansing, mi, usa, march 10-13, 2019, proceedings. In *Evolutionary Multi-Criterion Optimization*, pages 179–190. Springer International Publishing, 01 2019. ISBN 978-3-030-12597-4. doi: 10.1007/978-3-030-12598-1\_15.
- M. Emmerich, N. Hochstrate, and B. Naujoks. An emo algorithm using the hypervolume measure as selection criterion. *Lecture Notes in Computer Science*, 3410:62–76, 03 2005. doi: 10.1007/978-3-540-31880-4\_5.
- W. M. Ferreira, I. R. Meneghini, D. I. Brandao, and F. G. Guimarães. Preference cone based multi-objective evolutionary algorithm applied to optimal management of distributed energy resources in microgrids. *Applied Energy*, 274:115326, 2020. ISSN 0306-2619. doi: <https://doi.org/10.1016/j.apenergy.2020.115326>. URL <https://www.sciencedirect.com/science/article/pii/S0306261920308382>.
- M. Garza-Fabre, G. Toscano Pulido, and C. Coello. Ranking methods for many-objective optimization. *MICAI 2009. Lecture Notes in Computer Science*, 5845:633–645, 11 2009. doi: 10.1007/978-3-642-05258-3\_56.
- I. Giagkiozis, R. C. Purshouse, and P. J. Fleming. Towards understanding the cost of adaptation in decomposition-based optimization algorithms. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pages 615–620, 2013. doi: 10.1109/SMC.2013.110.
- L. He, H. Ishibuchi, A. Trivedi, and D. Srinivasan. Dynamic normalization in MOEA/d for multiobjective optimization. In *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, jul 2020. doi: 10.1109/cec48606.2020.9185849.
- Y. Hua, Y. Jin, and K. Hao. A clustering-based adaptive evolutionary algorithm for multi-objective optimization with irregular pareto fronts. *IEEE Transactions on Cybernetics*, 49(7):2758–2770, 2019. doi: 10.1109/TCYB.2018.2834466.

- Y. Hua, Q. Liu, K. Hao, and Y. Jin. A survey of evolutionary algorithms for multi-objective optimization problems with irregular pareto fronts. *IEEE/CAA Journal of Automatica Sinica*, 8(2):303–318, 2021. doi: 10.1109/JAS.2021.1003817.
- S. Huband, P. Hingston, L. Barone, and L. While. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, oct 2006. doi: 10.1109/tevc.2005.861417.
- H. Ishibuchi and T. Murata. A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 28(3):392–403, 1998. doi: 10.1109/5326.704576.
- H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Evolutionary many-objective optimization: A short review. *2008 IEEE Congress on Evolutionary Computation, CEC 2008*, pages 2419 – 2426, 07 2008. doi: 10.1109/CEC.2008.4631121.
- H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima. Simultaneous use of different scalarizing functions in moea/d. *Proceedings of the 12th Annual Genetic and Evolutionary Computation Conference, GECCO '10*, pages 519–526, 01 2010. doi: 10.1145/1830483.1830577.
- H. Ishibuchi, N. Akedo, H. Ohyanagi, and Y. Nojima. Behavior of emo algorithms on many-objective optimization problems with correlated objectives. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pages 1465–1472, 2011. doi: 10.1109/CEC.2011.5949788.
- H. Ishibuchi, N. Akedo, and Y. Nojima. Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. *Evolutionary Computation, IEEE Transactions on*, 19:264–283, 04 2015. doi: 10.1109/TEVC.2014.2315442.
- H. Ishibuchi, H. Masuda, and Y. Nojima. Pareto fronts of many-objective degenerate test problems. *IEEE Transactions on Evolutionary Computation*, 20(5):807–813, 2016. doi: 10.1109/TEVC.2015.2505784.
- H. Ishibuchi, K. Doi, and Y. Nojima. On the effect of normalization in moea/d for multi-objective and many-objective optimization. *Complex and Intelligent Systems*, 3, 10 2017a. doi: 10.1007/s40747-017-0061-9.
- H. Ishibuchi, Y. Setoguchi, H. Masuda, and Y. Nojima. Performance of decomposition-based many-objective algorithms strongly depends on pareto front shapes. *IEEE Transactions on Evolutionary Computation*, 21(2):169–190, 2017b. doi: 10.1109/TEVC.2016.2587749.

- H. Jain and K. Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18(4):602–622, 2014. doi: 10.1109/TEVC.2013.2281534.
- S. Jiang and S. Yang. An improved multiobjective optimization evolutionary algorithm based on decomposition for complex pareto fronts. *IEEE Transactions on Cybernetics*, 46:421–437, 02 2016. doi: 10.1109/TCYB.2015.2403131.
- S. Jiang and S. Yang. A strength pareto evolutionary algorithm based on reference direction for multi-objective and many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 21:329–346, 06 2017. doi: 10.1109/TEVC.2016.2592479.
- S. Jiang, S. Yang, Y. Wang, and X. Liu. Scalarizing functions in decomposition-based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 22(2):296–313, 2018. doi: 10.1109/TEVC.2017.2707980.
- J. R. Kasprzyk, P. M. Reed, G. W. Characklis, and B. R. Kirsch. Many-objective de novo water supply portfolio planning under deep uncertainty. *Environ. Model. Softw.*, 34: 87–104, 2012.
- B. Li, J. Li, K. Tang, and X. Yao. Many-objective evolutionary algorithms. *ACM Computing Surveys*, 48:1–35, 09 2015. doi: 10.1145/2792984.
- H. Li and D. Landa-Silva. An adaptive evolutionary multi-objective approach based on simulated annealing. *Evolutionary computation*, 19:561–95, 03 2011. doi: 10.1162/EVCO\_a\_00038.
- H. Li and Q. Zhang. Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii. *IEEE Trans. Evolutionary Computation*, 13:284–302, 04 2009. doi: 10.1109/TEVC.2008.925798.
- H. Li, M. Ding, J. Deng, and Q. Zhang. On the use of random weights in moea/d. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 978–985, 2015. doi: 10.1109/CEC.2015.7256996.
- K. Li, K. Deb, Q. Zhang, and S. Kwong. An evolutionary many-objective optimization algorithm based on dominance and decomposition. *IEEE Transactions on Evolutionary Computation*, 19(5):694–716, oct 2015. doi: 10.1109/tevc.2014.2373386.
- M. Li and X. Yao. What weights work for you? adapting weights for any pareto front shape in decomposition-based evolutionary multiobjective optimisation. *Evolutionary Computation*, 28(2):227–253, 2020. doi: 10.1162/evco\_a\_00269.



- M. Li, S. Yang, and X. Liu. Pareto or non-pareto: Bi-criterion evolution in multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 20(5):645–665, oct 2016. doi: 10.1109/tevc.2015.2504730.
- H.-L. Liu, F. Gu, and Q. Zhang. Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems. *IEEE Transactions on Evolutionary Computation*, 18(3):450–455, jun 2014. doi: 10.1109/tevc.2013.2281533.
- M. Luque, S. González-Gallardo, R. Saborido Infantes, and A. B. Ruiz. Adaptive global wasf-ga to handle many-objective optimization problems. *Swarm and Evolutionary Computation*, 54:100644, 05 2020. doi: 10.1016/j.swevo.2020.100644.
- X. Ma, Y. Yu, X. Li, Y. Qi, and Z. Zhu. A survey of weight vector adjustment methods for decomposition-based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 24(4):634–649, 2020. doi: 10.1109/TEVC.2020.2978158.
- I. Meneghini, F. Gadelha Guimarães, A. Gaspar-Cunha, and M. Weiss Cohen. *Incorporation of Region of Interest in a Decomposition-Based Multi-objective Evolutionary Algorithm*, pages 35–50. Springer International Publishing, Cham, 2021. ISBN 978-3-030-57422-2. doi: 10.1007/978-3-030-57422-2\_3. URL [https://doi.org/10.1007/978-3-030-57422-2\\_3](https://doi.org/10.1007/978-3-030-57422-2_3).
- I. R. Meneghini, M. A. Alves, A. Gaspar-Cunha, and F. G. Guimarães. Scalable and customizable benchmark problems for many-objective optimization. *Applied Soft Computing*, 90:106139, 2020a. ISSN 1568-4946. doi: <https://doi.org/10.1016/j.asoc.2020.106139>.
- I. R. Meneghini, F. G. Guimarães, A. Gaspar-Cunha, and M. W. Cohen. Incorporation of region of interest in a decomposition-based multi-objective evolutionary algorithm. In *Computational Methods in Applied Sciences*, pages 35–50. Springer International Publishing, nov 2020b. doi: 10.1007/978-3-030-57422-2\_3.
- Y. Qi, X. Ma, F. Liu, L. Jiao, J. Sun, and J. Wu. MOEA/d with adaptive weight adjustment. *Evolutionary Computation*, 22(2):231–264, jun 2014. doi: 10.1162/evco\_a\_00109.
- S. S. Rao. *Engineering Optimization Theory and Practice*, volume 4 of 10. The name of the publisher, The address, 3 edition, 7 2009. ISBN 3257227892. An optional note.
- H. Sato. Inverted pbi in moea/d and its impact on the search performance on multi and many-objective optimization. *GECCO 2014 - Proceedings of the 2014 Genetic and Evolutionary Computation Conference*, 07 2014. doi: 10.1145/2576768.2598297.
- L. Thiele, K. Miettinen, P. Korhonen, and J. Molina. A preference-based evolutionary algorithm for multi-objective optimization. *Evolutionary computation*, 17:411–36, 02 2009. doi: 10.1162/evco.2009.17.3.411.

- Y. Tian, R. Cheng, X. Zhang, and Y. Jin. PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]. *IEEE Computational Intelligence Magazine*, 12(4):73–87, nov 2017. doi: 10.1109/mci.2017.2742868.
- A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh. A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE Transactions on Evolutionary Computation*, PP:1–1, 09 2016. doi: 10.1109/TEVC.2016.2608507.
- D. A. V. Veldhuizen and G. B. Lamont. Multiobjective evolutionary algorithm research: A history and analysis, 1998.
- L. Wang and Q. Zhang. Constrained subproblems in a decomposition-based multiobjective evolutionary algorithm. *IEEE Transactions on Evolutionary Computation*, 20:1–1, 01 2015. doi: 10.1109/TEVC.2015.2457616.
- R. Wang, R. C. Purshouse, and P. J. Fleming. Preference-inspired co-evolutionary algorithms using weight vectors. *European Journal of Operational Research*, 243(2):423–441, 2015. ISSN 0377-2217. doi: <https://doi.org/10.1016/j.ejor.2014.05.019>.
- Z. Wang, Q. Zhang, M. Gong, and A. Zhou. A replacement strategy for balancing convergence and diversity in moea/d. *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*, pages 2132–2139, 07 2014. doi: 10.1109/CEC.2014.6900319.
- Y. Yuan, H. Xu, B. Wang, and X. Yao. A new dominance relation based evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 99:1–1, 01 2015. doi: 10.1109/TEVC.2015.2420112.
- Q. Zhang and H. Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007. doi: 10.1109/TEVC.2007.892759.
- Q. Zhang, W. Liu, and H. Li. The performance of a new version of MOEA/d on CEC09 unconstrained MOP test instances. In *2009 IEEE Congress on Evolutionary Computation*, page 203208. IEEE, may 2009. doi: 10.1109/cec.2009.4982949.
- A. Zhou and Q. Zhang. Are all the subproblems equally important? resource allocation in decomposition based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 20:1–1, 01 2015. doi: 10.1109/TEVC.2015.2424251.
- A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. Suganthan, and Q. Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1:32–49, 03 2011. doi: 10.1016/j.swevo.2011.03.001.

- 
- A. Zhou, Q. Zhang, and G. Zhang. A multiobjective evolutionary algorithm based on decomposition and probability model. *2012 IEEE Congress on Evolutionary Computation, CEC 2012*, pages 1–8, 06 2012. doi: 10.1109/CEC.2012.6252954.
- E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm. *TIK-Report*, 103, 07 2001.

# Apêndice A

## Problemas de Teste Propostos

Com o objetivo de contribuir com os problemas de fronteiras Pareto irregulares, são propostas duas novas funções de teste que contém características que exploram algumas dificuldades enfrentadas por algoritmos para obter uma distribuição uniforme de soluções. Assim, as novas funções de teste são implementadas usando o gerador de funções de teste escalável e customizável proposto em [Meneghini et al., 2020a], conhecido como gerador de *Generalized Position-Distance* (GPD).

O GPD é um método flexível e personalizável que gera problemas de teste com uma infinidade de características. A ideia principal do gerador é minimizar uma função (A.1) que é construída com dois componentes,  $F_p(x)$  e  $F_d(x)$ .

$$\begin{aligned} \min F(x) = F_p(x) \cdot F_d(x) \\ \text{Sujeito a restrições} \end{aligned} \tag{A.1}$$

A função  $F_p(x)$  controla a distribuição da geometria dos pontos no primeiro octante, controlando a posição relativa no espaço de objetivos. A função  $F_d(x)$  por outro lado, controla a convergência das soluções para a fronteira Pareto. O  $F_d(x)$  para as funções de teste propostas é construído usando a mesma função utilizada no problema de teste DTLZ2:

$$F_d(x) = \sum_{i=1}^M (x_i - 0.5)^2 \tag{A.2}$$

A primeira função de teste, chamada GPD01, é proposta com foco em uma fatia do simplex no primeiro octante do espaço de objetivos. O objetivo desta função de teste é apresentar dificuldade ao alterar a informação dos pontos extremos em comparação com

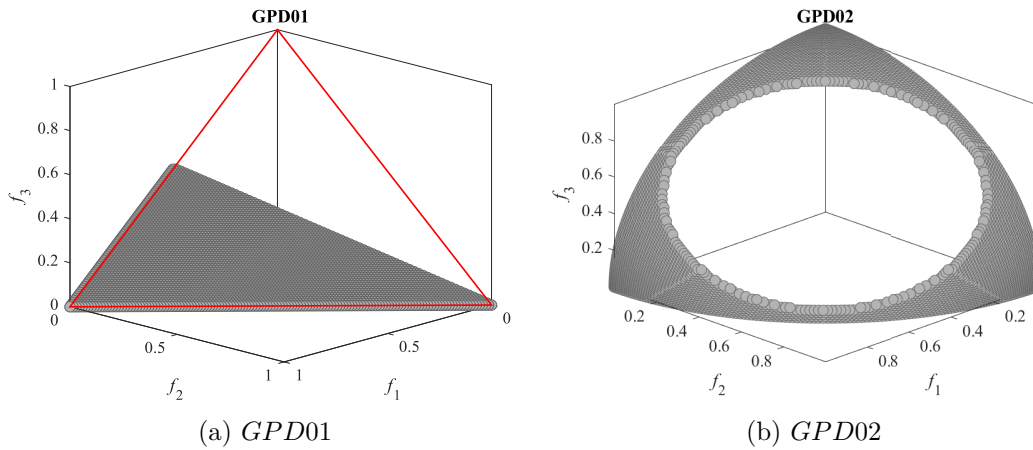


Figura 15 – Exemplos das fronteiras Pareto no espaço tridimensional das funções de teste propostas: a) GPD01 em comparação com a forma de um simplex regular em vermelho, b) GPD02 (Fonte: Própria).

a forma do simplex original. Para atingir este objetivo, o  $F_p(x)$  é definido como:

$$\begin{aligned}
 F_{p1}(x) &= \prod_{i=1}^{M-1} \cos\left(x_i \frac{\pi}{2}\right) \\
 F_{pM-1}(x) &= \prod_{j=2}^{M-j} \cos\left(x_j \frac{\pi}{2}\right) \times \sin\left(x_{j+1} \frac{\pi}{2}\right) \\
 F_{pM}(x) &= \sin\left(x_1 \frac{\pi}{2}\right)
 \end{aligned} \tag{A.3}$$

Na sequência, um ponto de corte é definido,  $c = \left\lfloor \frac{M}{2} \right\rfloor$ , que refletirá os pontos acima da linha do ponto de corte. Portanto, se  $\overline{F_{p(1:c)}(x)} < \overline{F_{p(M-c+1:M)}(x)}$ <sup>1</sup> os valores são trocados. Um exemplo da fronteira de Pareto para a função de teste GPD01 é apresentado na Figura 15a.

No caso da função GPD02, o objetivo é construir uma função teste que tenha semelhanças com uma fronteira desconectada. Para este propósito, o  $F_p(x)$  é construído da mesma maneira que (A.3), mas aqui um hiper-cone é considerado centrado na hiper-diagonal do espaço de objetivos e o ângulo de abertura dado por (3.11), onde  $\phi = 0,6$ , com todos os valores dentro do cone sendo penalizados. Um exemplo da geometria da fronteira Pareto da GPD02 em três dimensões é demonstrado na Figura 15b. A geometria pode ser referida como uma frente de Pareto com gênero 1, que consiste em uma superfície com um furo para todos  $M \geq 3$ .

<sup>1</sup> Média de  $F_p$  considerando o ponto de corte

## Apêndice B

### Tabelas de Resultados Completos de IGD

Tabela 9 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família DTLZ com o método de inicialização  $k$ -layers para grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEAD	MOEADDE	MOEADM2M	MOEADD	NSGAIII	MOEADLNA*	MOEADLNA
IDTLZ1	3	3.01e-2 (8.3e-5) -	2.84e-2 (4.3e-5) -	8.35e-1 (6.1e-1) -	2.87e-2 (7.2e-4) -	2.60e-2 (5.8e-4) -	3.01e-2 (3.9e-5) -	2.01e-2 (2.6e-4)
	5	2.30e-1 (2.5e-2) -	9.07e-2 (5.2e-4) -	1.87e+1 (9.7e+0) -	1.80e-1 (1.9e-2) -	7.41e-2 (3.7e-3) -	1.33e-1 (4.1e-3) -	5.93e-2 (6.6e-4)
	8	2.87e-1 (1.5e-2) -	1.52e-1 (1.7e-3) -	1.66e+2 (6.2e+1) -	5.16e-1 (1.4e-1) -	1.43e-1 (1.0e-2) -	1.72e-1 (1.5e-3) -	1.06e-1 (1.3e-3)
	10	2.93e-1 (1.4e-2) -	1.68e-1 (8.5e-3) -	1.74e+2 (5.1e+1) -	5.53e-1 (5.2e-1) -	1.53e-1 (1.2e-2) -	1.73e-1 (9.4e-4) -	1.22e-1 (1.9e-3)
IDTLZ2	3	6.86e-2 (4.7e-4) -	8.00e-2 (8.4e-4) -	1.15e-1 (7.5e-3) -	6.96e-2 (8.9e-4) -	7.08e-2 (2.9e-3) -	6.79e-2 (3.3e-4) -	5.89e-2 (1.4e-3)
	5	4.61e-1 (3.3e-3) -	2.70e-1 (8.2e-4) -	4.61e-1 (2.3e-2) -	4.59e-1 (1.7e-2) -	2.69e-1 (1.3e-2) -	4.68e-1 (1.5e-3) -	2.34e-1 (8.1e-3)
	8	6.94e-1 (7.4e-3) -	4.27e-1 (1.1e-3) =	7.70e-1 (4.4e-2) -	7.11e-1 (3.1e-2) -	4.56e-1 (7.9e-3) -	6.84e-1 (3.5e-3) -	4.29e-1 (8.2e-3)
	10	7.66e-1 (1.0e-2) -	4.71e-1 (1.2e-3) +	8.06e-1 (6.6e-2) -	9.36e-1 (2.3e-2) -	5.71e-1 (8.9e-3) -	7.29e-1 (2.5e-3) -	5.14e-1 (8.0e-3)
DTLZ1	3	1.86e-2 (6.3e-5) +	2.52e-2 (1.3e-4) -	7.52e-1 (4.7e-1) -	1.86e-2 (3.0e-5) +	1.86e-2 (4.9e-5) +	1.86e-2 (4.0e-5) +	1.89e-2 (1.5e-4)
	5	7.54e-2 (4.7e-5) -	1.38e-1 (2.0e-3) -	6.73e+0 (3.0e+0) -	7.54e-2 (4.4e-5) -	7.53e-2 (1.2e-4) -	7.53e-2 (6.9e-5) -	6.88e-2 (1.6e-3)
	8	1.16e-1 (1.3e-4) -	1.83e-1 (1.7e-3) -	1.21e+1 (5.2e+0) -	1.16e-1 (9.4e-5) -	1.31e-1 (3.5e-2) -	1.15e-1 (3.4e-4) -	1.08e-1 (2.6e-3)
	10	1.14e-1 (1.3e-4) =	1.76e-1 (2.7e-3) -	1.46e+1 (6.1e+0) -	1.13e-1 (1.4e-4) =	1.31e-1 (3.3e-2) =	1.45e-1 (5.4e-2) =	1.47e-1 (7.7e-2)
DTLZ2	3	4.86e-2 (4.7e-7) +	6.40e-2 (4.3e-4) -	9.92e-2 (7.8e-3) -	4.86e-2 (3.2e-5) +	4.86e-2 (2.2e-5) +	4.86e-2 (7.0e-7) +	4.95e-2 (3.0e-4)
	5	2.58e-1 (5.6e-5) -	3.84e-1 (1.2e-2) -	5.11e-1 (3.6e-2) -	2.56e-1 (2.1e-3) -	2.58e-1 (1.9e-4) -	2.58e-1 (8.7e-5) -	2.24e-1 (4.1e-3)
	8	4.32e-1 (2.2e-4) -	6.00e-1 (1.0e-2) -	7.80e-1 (3.3e-2) -	4.28e-1 (1.9e-3) -	4.33e-1 (1.0e-2) -	4.32e-1 (2.3e-4) -	3.74e-1 (3.3e-3)
	10	4.57e-1 (2.1e-4) -	6.56e-1 (1.4e-2) -	8.27e-1 (3.7e-2) -	4.56e-1 (8.4e-4) -	4.74e-1 (3.3e-2) -	4.57e-1 (2.4e-4) -	4.06e-1 (2.2e-3)
DTLZ3	3	5.10e-2 (1.9e-3) +	6.43e-2 (6.5e-4) -	3.52e+1 (2.0e+1) -	5.12e-2 (2.4e-3) +	5.17e-2 (4.3e-3) +	5.17e-2 (2.6e-3) +	5.30e-2 (2.5e-3)
	5	2.61e-1 (5.3e-3) -	4.31e-1 (8.2e-3) -	1.12e+2 (4.5e+1) -	2.69e-1 (6.0e-2) -	3.45e-1 (3.4e-1) =	2.61e-1 (2.5e-3) -	2.43e-1 (7.2e-3)
	8	4.57e-1 (1.0e-1) -	6.62e-1 (9.0e-3) -	2.06e+2 (3.9e+1) -	4.26e-1 (2.9e-3) -	1.30e+0 (1.5e+0) -	4.32e-1 (2.1e-3) -	3.99e-1 (1.6e-2)
	10	4.62e-1 (4.3e-3) -	7.11e-1 (1.4e-2) -	2.08e+2 (4.8e+1) -	4.60e-1 (2.5e-3) -	2.45e+0 (2.0e+0) -	4.61e-1 (3.9e-3) -	4.21e-1 (8.5e-3)
DTLZ4	3	4.23e-1 (2.4e-1) -	1.73e-1 (2.3e-1) -	6.33e-2 (2.9e-3) +	6.51e-2 (9.0e-2) +	9.79e-2 (1.5e-1) +	1.14e-1 (1.7e-1) +	1.18e-1 (1.7e-1)
	5	4.50e-1 (1.5e-1) -	3.86e-1 (5.1e-2) -	5.05e-1 (8.4e-2) -	2.56e-1 (2.0e-3) -	2.58e-1 (3.8e-4) -	2.78e-1 (6.1e-2) -	2.26e-1 (5.1e-2)
	8	5.86e-1 (1.1e-1) -	5.84e-1 (4.6e-2) -	9.92e-1 (1.4e-1) -	4.25e-1 (1.9e-3) -	4.34e-1 (2.4e-2) -	4.47e-1 (4.1e-2) -	3.76e-1 (2.8e-2)
	10	6.61e-1 (9.3e-2) -	6.62e-1 (3.8e-2) -	1.09e+0 (9.9e-2) -	4.57e-1 (1.3e-2) -	4.66e-1 (3.3e-2) -	4.80e-1 (3.3e-2) -	4.22e-1 (3.8e-2)
DTLZ5	3	1.50e-2 (7.7e-5) +	1.37e-2 (1.5e-4) +	1.21e-1 (4.1e-2) -	3.38e-2 (1.5e-3) -	1.01e-2 (1.4e-3) +	1.71e-2 (6.6e-4) +	2.15e-2 (5.2e-3)
	5	1.11e-2 (5.9e-5) +	9.33e-3 (9.1e-5) +	2.19e-1 (6.2e-2) -	7.77e-2 (5.6e-3) -	3.63e-2 (9.9e-3) +	1.26e-2 (1.5e-3) +	6.26e-2 (1.4e-2)
	8	1.17e-2 (1.1e-3) +	1.18e-2 (1.9e-5) +	3.40e-1 (8.4e-2) -	9.92e-2 (1.2e-2) -	6.41e-2 (2.2e-2) -	1.52e-2 (4.2e-3) +	3.64e-2 (1.2e-2)
	10	1.10e-2 (3.8e-4) -	1.21e-2 (2.2e-4) -	3.48e-1 (7.8e-2) -	1.17e-1 (1.1e-2) -	6.21e-2 (2.5e-2) -	1.40e-2 (5.0e-3) -	9.31e-3 (6.4e-3)
DTLZ6	3	1.51e-2 (6.8e-6) +	1.39e-2 (1.2e-4) +	9.12e-2 (1.8e-2) -	3.46e-2 (1.1e-3) =	1.57e-2 (2.0e-3) +	1.42e-2 (3.0e-4) +	4.20e-2 (3.0e-2)
	5	1.13e-2 (5.8e-4) +	9.21e-3 (1.6e-4) +	4.18e-1 (7.1e-1) -	7.70e-2 (7.7e-3) =	6.22e-2 (1.6e-2) +	9.21e-2 (5.1e-2) =	7.63e-2 (2.7e-2)
	8	1.10e-2 (1.6e-3) +	1.17e-2 (1.8e-4) +	2.10e+0 (8.7e-1) -	1.09e-1 (1.0e-2) =	1.03e-1 (3.3e-2) =	8.83e-2 (5.4e-2) =	1.12e-1 (4.7e-2)
	10	9.95e-3 (1.0e-3) +	1.19e-2 (2.9e-4) +	2.25e+0 (6.8e-1) -	1.20e-1 (8.7e-3) -	1.24e-1 (4.0e-2) -	7.72e-2 (1.2e-1) =	4.80e-2 (2.8e-2)
DTLZ7	3	1.54e-1 (1.2e-1) -	1.63e-1 (7.1e-3) -	9.03e-1 (1.0e+0) -	5.54e-1 (3.7e-1) -	7.04e-2 (2.8e-3) -	9.49e-2 (2.6e-3) -	6.23e-2 (1.9e-3)
	5	6.12e-1 (6.4e-2) -	6.80e-1 (7.6e-2) -	4.72e+0 (4.2e+0) -	1.28e+0 (6.5e-1) -	3.36e-1 (1.8e-2) -	5.38e-1 (6.1e-3) -	3.15e-1 (1.4e-2)
	8	1.84e+0 (2.6e-1) -	1.51e+0 (2.3e-1) -	2.42e+0 (7.7e+0) -	2.84e+0 (6.2e-1) -	9.04e-1 (6.0e-2) -	9.00e-1 (2.7e-3) -	6.18e-1 (7.3e-3)
	10	2.97e+0 (3.4e-1) -	2.41e+0 (4.1e-1) -	3.06e+1 (6.2e+0) -	3.06e+0 (9.0e-1) -	1.30e+0 (9.8e-2) -	1.09e+0 (5.4e-3) -	7.41e-1 (7.4e-3)
+/-/=		10/25/1	8/27/1	1/35/0	4/28/4	8/25/3	8/24/4	

Tabela 10 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família DTLZ com o método de inicialização  $k$ -layers para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA	MOEADLNA
IDTLZ1	3	2.40e-2 (7.6e-4) -	2.91e-2 (2.3e-4) -	7.83e-2 (1.0e-2) -	2.23e-2 (5.2e-4) -	4.50e-2 (2.6e-2) -	2.01e-2 (2.6e-4)
	5	1.31e-1 (2.0e-2) -	9.41e-2 (1.5e-3) -	1.24e-1 (1.2e-2) -	6.31e-2 (2.0e-3) -	2.61e-1 (2.3e-2) -	5.93e-2 (6.6e-4)
	8	2.29e-1 (3.4e-2) -	1.66e-1 (3.9e-3) -	1.46e-1 (1.4e-2) -	1.41e-1 (8.5e-3) -	3.30e-1 (1.5e-2) -	1.06e-1 (1.3e-3)
	10	2.10e-1 (3.6e-2) -	1.85e-1 (1.2e-2) -	1.54e-1 (1.5e-2) -	1.46e-1 (1.3e-2) -	3.50e-1 (2.0e-2) -	1.22e-1 (1.9e-3)
IDTLZ2	3	5.70e-2 (3.2e-3) +	8.57e-2 (1.2e-3) -	6.28e-2 (1.6e-3) -	7.53e-2 (6.0e-3) -	7.58e-2 (1.6e-3) -	5.89e-2 (1.4e-3)
	5	2.89e-1 (9.3e-3) -	2.70e-1 (1.1e-3) -	2.35e-1 (3.4e-3) =	2.70e-1 (2.0e-2) -	4.61e-1 (1.8e-2) -	2.34e-1 (8.1e-3)
	8	6.14e-1 (4.5e-2) -	4.32e-1 (2.7e-3) =	3.95e-1 (3.5e-3) +	3.80e-1 (9.5e-3) +	7.23e-1 (3.2e-2) -	4.29e-1 (8.2e-3)
	10	7.36e-1 (4.8e-2) -	4.80e-1 (4.0e-3) +	4.33e-1 (3.4e-3) +	5.70e-1 (7.7e-3) -	7.68e-1 (2.5e-2) -	5.14e-1 (8.0e-3)
DTLZ1	3	1.92e-2 (2.9e-4) -	2.57e-2 (5.4e-4) -	8.07e-2 (6.7e-3) -	2.53e-2 (1.2e-2) -	1.86e-2 (1.9e-5) +	1.89e-2 (1.5e-4)
	5	6.33e-2 (1.8e-3) +	1.39e-1 (2.4e-3) -	1.64e-1 (4.5e-2) -	7.16e-2 (1.6e-2) =	7.53e-2 (1.8e-4) -	6.88e-2 (1.6e-3)
	8	1.08e-1 (2.2e-3) =	1.76e-1 (5.8e-3) -	2.33e-1 (4.2e-2) -	1.16e-1 (1.6e-2) -	1.20e-1 (2.1e-2) -	1.08e-1 (2.6e-3)
	10	1.15e-1 (3.1e-3) =	1.67e-1 (4.1e-3) -	2.63e-1 (3.5e-2) -	1.55e-1 (6.1e-2) =	1.19e-1 (1.3e-2) =	1.47e-1 (7.7e-2)
DTLZ2	3	5.00e-2 (6.1e-4) -	6.44e-2 (5.3e-4) -	1.85e-1 (1.4e-2) -	5.39e-2 (1.8e-3) -	4.86e-2 (4.1e-5) +	4.95e-2 (3.0e-4)
	5	1.95e-1 (3.3e-3) +	4.40e-1 (1.0e-2) -	3.18e-1 (1.7e-2) -	2.20e-1 (5.6e-3) +	2.58e-1 (1.3e-4) -	2.24e-1 (4.1e-3)
	8	3.69e-1 (5.0e-3) +	6.67e-1 (1.4e-2) -	5.15e-1 (3.3e-2) -	3.98e-1 (4.6e-2) -	4.32e-1 (3.4e-4) -	3.74e-1 (3.3e-3)
	10	4.31e-1 (1.3e-2) -	6.95e-1 (1.9e-2) -	6.67e-1 (5.7e-2) -	5.56e-1 (5.8e-2) -	4.58e-1 (3.8e-4) -	4.06e-1 (2.2e-3)
DTLZ3	3	5.31e-2 (1.7e-3) =	1.27e-1 (9.5e-2) -	2.23e-1 (1.3e-2) -	6.60e-2 (9.9e-3) -	5.19e-2 (3.1e-3) +	5.30e-2 (2.5e-3)
	5	2.22e-1 (9.9e-3) +	4.78e-1 (5.7e-2) -	4.13e-1 (2.7e-2) -	3.64e-1 (4.1e-1) =	4.13e-1 (3.8e-1) -	2.43e-1 (7.2e-3)
	8	4.29e-1 (4.7e-2) -	6.79e-1 (2.8e-2) -	7.13e-1 (5.1e-2) -	1.30e+0 (7.6e-1) -	4.76e-1 (1.0e-1) -	3.99e-1 (1.6e-2)
	10	5.26e-1 (8.7e-2) -	7.17e-1 (4.3e-2) -	8.22e-1 (5.8e-2) -	2.35e+0 (2.1e+0) -	5.32e-1 (6.0e-2) -	4.21e-1 (8.5e-3)
DTLZ4	3	2.62e-1 (2.7e-1) =	9.95e-2 (1.2e-1) +	4.01e-1 (2.6e-1) -	1.48e-1 (2.0e-1) -	4.86e-2 (6.3e-5) +	1.18e-1 (1.7e-1)
	5	3.95e-1 (1.7e-1) -	5.22e-1 (5.7e-2) -	4.64e-1 (1.6e-1) -	2.36e-1 (4.5e-2) -	2.58e-1 (1.8e-4) -	2.26e-1 (5.1e-2)
	8	5.67e-1 (1.0e-1) -	7.80e-1 (6.0e-2) -	6.26e-1 (7.8e-2) -	4.06e-1 (5.4e-2) -	4.31e-1 (5.9e-4) -	3.76e-1 (2.8e-2)
	10	5.64e-1 (9.9e-2) -	8.41e-1 (6.8e-2) -	6.62e-1 (6.5e-2) -	4.43e-1 (3.5e-2) -	4.57e-1 (6.3e-4) -	4.22e-1 (3.8e-2)
DTLZ5	3	6.58e-3 (2.3e-4) +	1.38e-2 (9.1e-5) +	3.11e-2 (8.9e-3) -	1.32e-2 (2.6e-3) +	3.80e-2 (7.0e-4) -	2.15e-2 (5.2e-3)
	5	1.07e-2 (3.8e-3) +	9.17e-3 (1.9e-4) +	5.44e-2 (1.0e-2) +	3.86e-2 (9.8e-3) +	2.36e-1 (1.3e-2) -	6.26e-2 (1.4e-2)
	8	2.11e-2 (6.8e-3) +	1.15e-2 (3.1e-4) +	5.92e-2 (1.2e-2) -	1.31e-1 (9.2e-2) -	3.44e-1 (4.4e-2) -	3.64e-2 (1.2e-2)
	10	1.58e-2 (7.0e-3) -	1.15e-2 (6.5e-4) -	5.24e-2 (9.6e-3) -	1.26e-1 (7.0e-2) -	4.20e-1 (6.5e-2) -	9.31e-3 (6.4e-3)
DTLZ6	3	6.85e-3 (4.1e-4) +	1.38e-2 (1.1e-4) +	4.58e-2 (1.3e-2) =	1.84e-2 (3.7e-3) +	3.70e-2 (1.9e-3) =	4.20e-2 (3.0e-2)
	5	1.48e-2 (8.9e-3) +	8.92e-3 (6.7e-5) +	8.44e-2 (1.3e-2) =	7.34e-2 (2.6e-2) =	1.88e-1 (2.8e-2) -	7.63e-2 (2.7e-2)
	8	2.29e-2 (1.3e-2) +	1.07e-2 (6.0e-4) +	7.68e-2 (1.3e-2) +	2.24e-1 (1.1e-1) -	3.73e-1 (5.5e-2) -	1.12e-1 (4.7e-2)
	10	3.11e-2 (1.7e-2) +	1.01e-2 (6.2e-4) +	7.83e-2 (1.6e-2) -	3.35e-1 (2.0e-1) -	4.41e-1 (6.1e-2) -	4.80e-2 (2.8e-2)
DTLZ7	3	1.27e-1 (1.1e-1) -	4.35e-1 (2.8e-1) -	1.32e-1 (6.4e-2) -	8.04e-2 (5.6e-2) -	9.22e-2 (1.3e-3) -	6.23e-2 (1.9e-3)
	5	4.35e-1 (6.1e-2) -	1.15e+0 (2.0e-1) -	4.55e-1 (2.7e-2) -	3.62e-1 (1.8e-2) -	4.81e-1 (1.2e-2) -	3.15e-1 (1.4e-2)
	8	1.04e+0 (8.2e-2) -	1.82e+0 (5.0e-1) -	1.18e+0 (1.0e-1) -	9.23e-1 (6.8e-2) -	9.10e-1 (3.7e-2) -	6.18e-1 (7.3e-3)
	10	1.43e+0 (2.2e-1) -	1.99e+0 (5.2e-1) -	1.60e+0 (9.5e-2) -	1.28e+0 (1.1e-1) -	1.16e+0 (3.1e-2) -	7.41e-1 (7.4e-3)
+/-/=		12/20/4	9/26/1	4/29/3	5/27/4	4/30/2	



Tabela 11 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família DTLZ com o método de inicialização Riesz S-Energy para grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEAD	MOEADDE	MOEADM2M	MOEADD	NSGAIII	MOEADLNA*	MOEADLNA
IDTLZ1	3	3.07e-2 (5.1e-5) -	2.93e-2 (5.7e-5) -	7.85e-1 (6.9e-1) -	3.23e-2 (1.3e-3) -	2.63e-2 (7.2e-4) -	3.07e-2 (2.6e-5) -	1.90e-2 (2.5e-4)
	5	1.30e-1 (3.3e-2) -	8.12e-2 (2.0e-4) -	1.44e+1 (8.9e+0) -	1.25e-1 (1.1e-2) -	1.11e-1 (5.2e-3) -	6.87e-2 (1.5e-4) -	6.07e-2 (6.9e-4)
	8	2.96e-1 (2.5e-2) -	1.10e-1 (4.8e-4) -	4.67e+1 (3.2e+1) -	1.84e-1 (1.1e-2) -	1.12e-1 (2.8e-3) -	1.43e-1 (6.7e-4) -	1.07e-1 (1.4e-3)
	10	3.28e-1 (1.9e-2) -	1.21e-1 (4.6e-4) +	2.86e+1 (2.3e+1) -	1.86e-1 (9.0e-3) -	1.19e-1 (1.6e-3) +	1.60e-1 (1.9e-3) -	1.24e-1 (2.5e-3)
IDTLZ2	3	6.25e-2 (6.9e-4) -	7.73e-2 (1.2e-3) -	1.00e-1 (6.4e-3) -	6.48e-2 (1.2e-3) -	6.47e-2 (2.6e-3) -	6.21e-2 (6.0e-4) -	5.55e-2 (1.7e-3)
	5	2.44e-1 (2.5e-4) -	2.00e-1 (1.1e-3) =	4.23e-1 (1.9e-2) -	2.49e-1 (7.4e-3) -	2.25e-1 (7.5e-3) -	2.44e-1 (3.0e-4) -	1.98e-1 (4.9e-3)
	8	5.47e-1 (1.7e-3) -	3.37e-1 (9.5e-4) +	6.34e-1 (2.0e-2) -	5.53e-1 (1.7e-2) -	4.16e-1 (1.0e-2) -	5.45e-1 (7.6e-4) -	4.11e-1 (5.4e-3)
	10	6.53e-1 (9.6e-3) -	4.07e-1 (1.0e-3) +	6.49e-1 (2.4e-2) -	6.46e-1 (2.8e-3) -	4.69e-1 (7.0e-3) +	6.34e-1 (1.4e-3) -	5.09e-1 (5.4e-3)
DTLZ1	3	1.77e-2 (7.3e-5) =	2.80e-2 (3.7e-4) -	8.51e-1 (6.0e-1) -	1.77e-2 (3.9e-5) =	1.77e-2 (6.1e-5) =	1.77e-2 (4.1e-5) =	1.77e-2 (8.8e-5)
	5	5.82e-2 (2.0e-4) -	1.60e+0 (9.7e-1) -	7.61e+0 (4.1e+0) -	5.80e-2 (1.2e-4) -	5.84e-2 (3.5e-4) -	5.77e-2 (2.4e-4) =	5.79e-2 (3.4e-4)
	8	9.75e-2 (5.5e-4) =	3.11e+1 (3.5e+1) -	8.46e+0 (4.4e+0) -	9.82e-2 (4.1e-4) =	1.06e-1 (2.2e-2) -	1.25e-1 (7.7e-2) =	1.05e-1 (1.7e-2)
	10	1.05e-1 (7.9e-4) +	5.98e+1 (3.1e+1) -	9.54e+0 (3.2e+0) -	1.07e-1 (8.9e-4) +	1.32e-1 (3.9e-2) +	2.18e-1 (8.7e-2) =	1.96e-1 (8.9e-2)
DTLZ2	3	4.67e-2 (4.6e-7) +	6.41e-2 (3.2e-4) -	8.48e-2 (6.6e-3) -	4.67e-2 (7.8e-6) +	4.68e-2 (2.1e-5) +	4.67e-2 (6.7e-7) +	4.76e-2 (3.2e-4)
	5	1.81e-1 (7.1e-6) +	4.33e-1 (1.2e-2) -	4.68e-1 (3.0e-2) -	1.81e-1 (6.4e-6) +	1.81e-1 (2.8e-5) +	1.81e-1 (7.3e-6) +	1.81e-1 (4.5e-4)
	8	3.32e-1 (1.9e-5) +	6.22e-1 (1.9e-2) -	6.81e-1 (2.7e-2) -	3.32e-1 (1.9e-5) +	3.37e-1 (2.8e-2) -	3.32e-1 (2.7e-5) +	3.34e-1 (6.7e-4)
	10	4.07e-1 (6.8e-5) -	6.70e-1 (2.1e-2) -	7.17e-1 (1.8e-2) -	4.07e-1 (5.5e-5) -	4.15e-1 (2.5e-2) -	4.07e-1 (4.1e-5) -	4.02e-1 (6.7e-4)
DTLZ3	3	4.94e-2 (2.6e-3) +	6.45e-2 (7.6e-4) -	3.47e+1 (1.8e+1) -	6.34e-2 (5.7e-2) =	5.69e-2 (3.3e-2) =	4.97e-2 (2.8e-3) =	5.09e-2 (5.2e-3)
	5	2.15e-1 (1.7e-1) =	5.47e-1 (1.3e-1) -	1.22e+2 (5.2e+1) -	1.97e-1 (6.8e-2) =	2.97e-1 (4.1e-1) -	1.84e-1 (8.2e-3) =	1.84e-1 (2.8e-3)
	8	6.16e-1 (3.5e-1) -	6.08e+0 (4.3e+0) -	2.21e+2 (3.4e+1) -	3.65e-1 (1.5e-1) -	2.36e+0 (2.0e+0) -	3.36e-1 (3.7e-3) =	3.35e-1 (1.8e-3)
	10	7.59e-1 (3.4e-1) -	1.84e+1 (1.1e+1) -	2.25e+2 (1.9e+1) -	4.12e-1 (7.5e-3) -	4.73e+0 (3.6e+0) -	4.06e-1 (2.6e-3) =	4.06e-1 (2.2e-3)
DTLZ4	3	2.25e-1 (2.7e-1) -	1.27e-1 (1.7e-1) -	6.08e-2 (2.1e-3) +	6.33e-2 (9.1e-2) +	9.62e-2 (1.5e-1) +	7.97e-2 (1.3e-1) +	1.13e-1 (2.0e-1)
	5	3.52e-1 (1.5e-1) -	4.87e-1 (5.7e-2) -	3.26e-1 (4.0e-2) -	1.81e-1 (1.8e-5) +	1.90e-1 (4.6e-2) +	1.89e-1 (4.3e-2) +	2.04e-1 (9.1e-2)
	8	5.33e-1 (9.2e-2) -	6.89e-1 (4.4e-2) -	9.49e-1 (1.2e-1) -	3.32e-1 (4.4e-5) +	3.38e-1 (2.8e-2) +	3.45e-1 (3.5e-2) +	3.60e-1 (3.9e-2)
	10	6.05e-1 (7.7e-2) -	7.16e-1 (5.0e-2) -	1.02e+0 (9.4e-2) -	4.07e-1 (1.1e-4) +	4.20e-1 (3.7e-2) -	4.15e-1 (2.8e-2) +	4.16e-1 (2.3e-2)
DTLZ5	3	2.73e-2 (1.6e-5) -	1.05e-2 (2.7e-4) +	6.76e-2 (1.5e-2) -	2.62e-2 (1.0e-3) -	8.82e-3 (9.9e-4) +	1.50e-2 (5.7e-4) +	1.94e-2 (3.1e-3)
	5	2.17e-2 (3.6e-5) +	3.74e-2 (1.4e-3) +	3.94e-1 (6.5e-2) -	9.52e-2 (2.3e-2) =	8.89e-2 (2.7e-2) =	2.07e-2 (1.4e-3) +	8.35e-2 (2.1e-2)
	8	2.78e-2 (6.0e-6) -	6.95e-2 (3.2e-4) -	5.54e-1 (1.3e-1) -	1.16e-1 (4.5e-3) -	2.85e-1 (7.1e-2) -	7.90e-2 (8.9e-2) -	1.73e-2 (1.5e-2)
	10	4.29e-2 (6.9e-6) -	8.74e-2 (8.5e-5) -	6.91e-1 (1.7e-1) -	1.20e-1 (5.9e-3) -	2.65e-1 (5.2e-2) -	1.82e-1 (1.9e-1) -	3.37e-2 (5.4e-2)
DTLZ6	3	2.73e-2 (4.8e-6) +	1.09e-2 (3.0e-4) +	7.41e-2 (1.3e-2) -	2.74e-2 (6.4e-4) +	1.32e-2 (1.7e-3) +	1.54e-2 (1.4e-3) +	5.01e-2 (2.7e-2)
	5	2.08e-2 (3.5e-3) +	3.71e-2 (7.6e-4) +	4.79e-1 (3.9e-1) -	9.31e-2 (2.6e-2) =	9.69e-2 (2.7e-2) =	1.95e-2 (6.6e-3) +	1.14e-1 (5.7e-2)
	8	2.78e-2 (4.6e-6) +	6.91e-2 (7.8e-4) +	2.30e+0 (6.5e-1) -	1.18e-1 (1.2e-2) +	3.04e-1 (1.9e-1) +	6.26e-1 (6.1e-2) -	5.28e-1 (1.2e-1)
	10	4.29e-2 (4.9e-6) +	8.72e-2 (5.1e-4) +	2.47e+0 (4.8e-1) -	1.23e-1 (6.8e-3) +	2.86e-1 (9.2e-2) +	5.81e-1 (1.6e-1) =	5.84e-1 (1.1e-1)
DTLZ7	3	1.64e-1 (1.7e-1) -	1.85e-1 (1.4e-2) -	1.57e+0 (2.3e+0) -	3.97e-1 (1.5e-1) -	6.44e-2 (2.0e-3) -	8.90e-2 (2.0e-3) -	5.87e-2 (1.6e-3)
	5	5.62e-1 (2.3e-2) -	7.10e-1 (7.1e-2) -	1.68e+1 (3.6e+0) -	3.00e+0 (8.7e-16) -	3.09e-1 (1.1e-2) -	3.84e-1 (2.3e-3) -	2.86e-1 (7.0e-3)
	8	2.93e+0 (1.1e+0) -	1.31e+0 (9.8e-2) -	2.70e+1 (5.0e+0) -	4.23e+0 (9.4e-1) -	6.78e-1 (9.6e-3) -	7.05e-1 (2.7e-3) -	6.22e-1 (8.8e-3)
	10	4.18e+0 (1.4e+0) -	1.44e+0 (7.6e-2) -	3.13e+1 (6.8e+0) -	8.74e+0 (2.2e-1) -	9.60e-1 (3.7e-2) -	7.59e-1 (3.4e-3) -	7.19e-1 (6.9e-3)
+/-/=		10/23/3	9/26/1	1/35/0	11/19/6	12/20/4	11/16/9	

Tabela 12 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família DTLZ com o método de inicialização Riesz S-Energy para grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA	MOEADLNA
IDTLZ1	3	3.04e-2 (1.6e-2) -	3.06e-2 (6.4e-4) -	8.39e-2 (1.0e-2) -	2.02e-2 (5.3e-4) -	3.71e-2 (7.6e-3) -	1.90e-2 (2.5e-4)
	5	3.12e+1 (1.2e+1) -	8.64e-2 (1.7e-3) -	1.26e-1 (1.2e-2) -	7.26e-2 (2.4e-3) -	1.48e-1 (1.8e-2) -	6.07e-2 (6.9e-4)
	8	6.32e+1 (1.8e+1) -	1.15e-1 (1.2e-3) -	1.47e-1 (1.0e-2) -	1.13e-1 (2.8e-3) -	2.57e-1 (1.6e-2) -	1.07e-1 (1.4e-3)
	10	7.12e+1 (2.3e+1) -	1.20e-1 (9.8e-4) +	1.57e-1 (8.7e-3) -	1.19e-1 (1.8e-3) +	2.70e-1 (2.3e-2) -	1.24e-1 (2.5e-3)
IDTLZ2	3	6.05e-2 (4.5e-3) -	8.59e-2 (1.4e-3) -	5.79e-2 (1.3e-3) -	6.68e-2 (3.9e-3) -	7.24e-2 (7.7e-4) -	5.55e-2 (1.7e-3)
	5	4.59e-1 (4.5e-2) -	2.11e-1 (2.2e-3) -	1.97e-1 (2.2e-3) =	2.22e-1 (8.0e-3) -	2.84e-1 (4.7e-3) -	1.98e-1 (4.9e-3)
	8	9.60e-1 (1.5e-1) -	3.77e-1 (3.7e-3) +	3.55e-1 (2.7e-3) +	4.16e-1 (9.8e-3) -	5.57e-1 (3.8e-3) -	4.11e-1 (5.4e-3)
	10	1.05e+0 (1.4e-1) -	4.30e-1 (2.4e-3) +	4.15e-1 (2.2e-3) +	4.72e-1 (9.0e-3) +	6.29e-1 (4.5e-3) -	5.09e-1 (5.4e-3)
DTLZ1	3	2.22e-2 (3.7e-3) -	2.90e-2 (1.4e-3) -	8.69e-2 (6.2e-3) -	2.05e-2 (2.2e-3) -	1.77e-2 (5.8e-5) +	1.77e-2 (8.8e-5)
	5	1.31e+1 (4.5e+0) -	3.06e+0 (4.0e+0) -	3.14e-1 (1.9e-1) -	6.28e-2 (6.9e-3) -	5.80e-2 (1.2e-4) -	5.79e-2 (3.4e-4)
	8	2.74e+1 (1.2e+1) -	3.24e+1 (6.0e+0) -	4.16e+0 (4.0e+0) -	1.02e-1 (6.2e-3) +	9.90e-2 (4.9e-4) =	1.05e-1 (1.7e-2)
	10	2.36e+1 (1.1e+1) -	2.59e+1 (1.0e+1) -	8.66e+0 (6.3e+0) -	1.38e-1 (4.3e-2) +	1.11e-1 (7.4e-3) +	1.96e-1 (8.9e-2)
DTLZ2	3	5.04e-2 (9.7e-4) -	6.55e-2 (8.4e-4) -	1.84e-1 (1.4e-2) -	4.85e-2 (1.3e-3) -	4.68e-2 (7.3e-5) +	4.76e-2 (3.2e-4)
	5	5.52e-1 (5.4e-2) -	5.62e-1 (2.6e-2) -	3.40e-1 (8.4e-3) -	1.82e-1 (5.5e-4) -	1.81e-1 (2.4e-5) +	1.81e-1 (4.5e-4)
	8	9.89e-1 (8.3e-2) -	1.04e+0 (7.7e-2) -	4.54e-1 (7.1e-3) -	3.32e-1 (9.0e-5) +	3.32e-1 (5.9e-5) +	3.34e-1 (6.7e-4)
	10	1.05e+0 (9.5e-2) -	1.14e+0 (9.6e-2) -	5.31e-1 (8.6e-3) -	4.23e-1 (3.9e-2) -	4.07e-1 (1.2e-4) -	4.02e-1 (6.7e-4)
DTLZ3	3	1.44e+0 (2.1e+0) -	1.72e+0 (4.5e+0) -	2.23e-1 (1.2e-2) -	6.46e-2 (1.4e-2) -	8.53e-2 (1.8e-1) -	5.09e-2 (5.2e-3)
	5	2.28e+2 (5.3e+1) -	1.80e+1 (1.8e+1) -	4.16e-1 (1.0e-2) -	3.44e-1 (4.1e-1) -	1.85e-1 (3.4e-3) =	1.84e-1 (2.8e-3)
	8	3.23e+2 (8.0e+1) -	1.39e+2 (5.6e+1) -	6.55e+0 (4.7e+0) -	2.15e+0 (1.8e+0) -	3.44e-1 (3.0e-2) -	3.35e-1 (1.8e-3)
	10	3.16e+2 (8.3e+1) -	1.79e+2 (3.9e+1) -	2.15e+1 (1.5e+1) -	4.25e+0 (4.2e+0) -	4.92e-1 (1.1e-1) -	4.06e-1 (2.2e-3)
DTLZ4	3	3.23e-1 (2.9e-1) -	1.19e-1 (1.0e-1) -	3.43e-1 (2.7e-1) -	9.70e-2 (1.5e-1) +	4.67e-2 (3.8e-6) +	1.13e-1 (2.0e-1)
	5	6.67e-1 (7.6e-2) -	6.64e-1 (4.3e-2) -	3.65e-1 (1.4e-1) -	1.90e-1 (4.3e-2) =	1.81e-1 (3.4e-5) +	2.04e-1 (9.1e-2)
	8	1.07e+0 (7.5e-2) -	1.06e+0 (7.2e-2) -	4.71e-1 (7.6e-2) -	3.49e-1 (5.0e-2) +	3.32e-1 (7.9e-5) +	3.60e-1 (3.9e-2)
	10	1.17e+0 (8.3e-2) -	1.25e+0 (9.8e-2) -	5.35e-1 (4.5e-2) -	4.13e-1 (2.6e-2) +	4.07e-1 (1.6e-4) +	4.16e-1 (2.3e-2)
DTLZ5	3	6.37e-3 (2.5e-4) +	1.08e-2 (2.4e-4) +	3.06e-2 (7.1e-3) -	9.61e-3 (1.6e-3) +	6.07e-2 (1.3e-3) -	1.94e-2 (3.1e-3)
	5	8.55e-2 (1.8e-2) =	3.69e-2 (1.0e-3) +	7.12e-2 (1.9e-2) +	1.13e-1 (5.6e-2) -	2.04e-1 (1.3e-2) -	8.35e-2 (2.1e-2)
	8	1.31e-1 (3.9e-2) -	6.80e-2 (2.8e-3) -	9.68e-2 (2.7e-2) -	2.97e-1 (7.9e-2) -	1.44e-1 (2.7e-2) -	1.73e-2 (1.5e-2)
	10	1.35e-1 (4.6e-2) -	8.70e-2 (7.5e-4) -	8.92e-2 (2.5e-2) -	2.80e-1 (6.4e-2) -	1.39e-1 (1.9e-2) -	3.37e-2 (5.4e-2)
DTLZ6	3	6.60e-3 (4.6e-4) +	1.10e-2 (5.7e-5) +	4.82e-2 (9.4e-3) =	1.27e-2 (2.0e-3) +	5.99e-2 (6.0e-3) -	5.01e-2 (2.7e-2)
	5	2.78e+0 (4.3e-1) -	3.56e-2 (1.0e-3) +	1.07e-1 (1.9e-2) =	1.61e-1 (7.7e-2) -	1.64e-1 (2.8e-2) -	1.14e-1 (5.7e-2)
	8	3.61e+0 (5.6e-1) -	6.86e-2 (7.8e-4) +	1.40e-1 (1.5e-2) +	3.17e-1 (2.1e-1) +	1.40e-1 (3.3e-2) +	5.28e-1 (1.2e-1)
	10	3.39e+0 (6.5e-1) -	8.71e-2 (4.0e-4) +	1.31e-1 (2.1e-2) +	3.04e-1 (1.7e-1) +	1.40e-1 (3.4e-2) +	5.84e-1 (1.1e-1)
DTLZ7	3	1.27e-1 (9.3e-2) -	2.43e-1 (7.0e-2) -	1.13e-1 (6.7e-3) -	6.54e-2 (2.7e-3) -	9.23e-2 (2.6e-3) -	5.87e-2 (1.6e-3)
	5	5.72e+0 (1.5e+0) -	1.60e+0 (3.0e-1) -	3.51e-1 (6.2e-2) -	3.08e-1 (9.2e-3) -	4.17e-1 (5.1e-3) -	2.86e-1 (7.0e-3)
	8	2.48e+1 (4.0e+0) -	4.69e+0 (4.5e-1) -	7.71e-1 (7.5e-2) -	6.84e-1 (9.2e-3) -	1.02e+0 (4.0e-2) -	6.22e-1 (8.8e-3)
	10	3.22e+1 (5.5e+0) -	6.81e+0 (6.2e-1) -	1.05e+0 (2.7e-2) -	9.65e-1 (3.3e-2) -	1.50e+0 (1.6e-1) -	7.19e-1 (6.9e-3)
+/-/=		2/33/1	9/27/0	5/28/3	12/23/1	11/23/2	

Tabela 13 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família WFG com o método de inicialização  $k$ -layers para grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEAD	MOEADDE	MOEADM2M	MOEADD	NSGAIII	MOEADLNA*	MOEADLNA
WFG1	3	2.54e-1 (1.3e-2) =	2.33e-1 (7.8e-3) =	1.55e+0 (1.1e-2) -	2.22e-1 (4.7e-2) +	<b>1.46e-1 (7.7e-3) +</b>	2.35e-1 (3.0e-2) =	2.47e-1 (3.5e-2)
	5	6.66e-1 (1.2e-2) -	1.01e+0 (3.3e-2) -	2.08e+0 (2.4e-2) -	9.32e-1 (1.4e-1) -	<b>6.20e-1 (7.4e-2) =</b>	6.23e-1 (2.4e-2) =	6.20e-1 (2.3e-2)
	8	1.43e+0 (3.2e-2) +	1.74e+0 (3.4e-2) -	2.67e+0 (2.1e-2) -	1.94e+0 (1.8e-1) -	<b>1.03e+0 (4.6e-2) +</b>	1.39e+0 (2.5e-2) +	1.48e+0 (5.1e-2)
	10	1.58e+0 (6.8e-2) +	1.85e+0 (2.7e-2) -	3.03e+0 (2.4e-2) -	2.51e+0 (2.1e-1) -	<b>1.15e+0 (6.6e-2) +</b>	1.58e+0 (2.3e-2) +	1.62e+0 (3.3e-2)
WFG2	3	2.33e-1 (3.5e-3) -	2.15e-1 (1.7e-3) -	2.38e-1 (1.2e-2) -	1.72e-1 (4.7e-3) +	<b>1.48e-1 (1.1e-3) +</b>	1.67e-1 (1.3e-2) +	1.94e-1 (1.8e-2)
	5	6.95e-1 (9.4e-3) -	1.03e+0 (5.4e-2) -	5.38e-1 (1.8e-2) +	5.29e-1 (3.8e-3) +	<b>4.93e-1 (4.7e-3) +</b>	6.35e-1 (8.3e-3) -	6.25e-1 (1.5e-2)
	8	1.65e+0 (2.6e-2) +	1.86e+0 (1.7e-1) -	1.06e+0 (3.3e-2) +	1.11e+0 (1.6e-2) +	<b>9.79e-1 (5.1e-2) +</b>	1.59e+0 (3.8e-2) +	1.69e+0 (5.6e-2)
	10	1.77e+0 (2.0e-2) +	1.97e+0 (1.4e-1) -	1.26e+0 (7.9e-2) +	1.18e+0 (1.6e-2) +	<b>1.17e+0 (1.7e-1) +</b>	1.74e+0 (3.1e-2) +	1.81e+0 (4.4e-2)
WFG3	3	1.48e-1 (4.2e-2) -	<b>7.21e-2 (6.3e-4) +</b>	3.31e-1 (3.1e-2) -	1.86e-1 (5.3e-2) -	9.74e-2 (1.1e-2) +	8.98e-2 (2.7e-3) +	1.22e-1 (1.6e-2)
	5	4.58e-1 (7.5e-2) =	<b>1.35e-1 (1.2e-3) +</b>	7.85e-1 (1.4e-1) -	1.01e+0 (6.5e-2) -	3.87e-1 (4.2e-2) +	2.16e-1 (9.9e-3) +	4.53e-1 (5.0e-2)
	8	1.74e+0 (2.7e-1) -	<b>5.68e-1 (1.2e-2) +</b>	1.54e+0 (2.6e-1) -	2.05e+0 (7.7e-2) -	8.38e-1 (1.8e-1) +	7.64e-1 (4.3e-2) +	1.09e+0 (1.1e-1)
	10	3.16e+0 (2.0e-1) -	7.37e-1 (1.6e-2) +	1.97e+0 (3.0e-1) -	2.61e+0 (9.9e-2) -	<b>3.92e-1 (5.6e-2) +</b>	1.17e+0 (9.9e-2) +	1.52e+0 (1.5e-1)
WFG4	3	2.32e-1 (3.9e-3) =	2.86e-1 (4.0e-3) -	3.06e-1 (1.1e-2) -	2.20e-1 (2.3e-3) +	<b>1.93e-1 (2.6e-4) +</b>	2.06e-1 (4.0e-3) +	2.32e-1 (6.3e-3)
	5	1.56e+0 (3.1e-2) -	2.50e+0 (2.1e-1) -	1.40e+0 (3.8e-2) -	1.49e+0 (6.6e-3) -	1.45e+0 (3.0e-2) -	1.46e+0 (3.1e-3) -	<b>1.12e+0 (1.8e-2)</b>
	8	5.18e+0 (1.8e-1) -	5.19e+0 (3.9e-1) -	3.57e+0 (6.2e-2) -	3.80e+0 (4.9e-2) -	3.70e+0 (1.3e-1) -	3.79e+0 (8.5e-3) -	<b>3.02e+0 (4.0e-2)</b>
	10	9.43e+0 (4.6e-1) -	6.72e+0 (5.0e-1) -	4.99e+0 (7.3e-2) =	5.78e+0 (9.2e-2) =	<b>4.95e+0 (1.4e-1) =</b>	5.02e+0 (1.2e-2) =	5.57e+0 (1.7e+0)
WFG5	3	2.30e-1 (2.0e-3) -	3.04e-1 (1.1e-3) -	2.84e-1 (1.2e-2) -	2.29e-1 (3.3e-3) -	<b>2.06e-1 (1.5e-4) +</b>	2.09e-1 (2.0e-3) +	2.23e-1 (4.7e-3)
	5	1.54e+0 (2.3e-2) -	2.55e+0 (7.4e-2) -	1.35e+0 (4.5e-2) -	1.49e+0 (7.6e-3) -	1.45e+0 (1.1e-2) -	1.45e+0 (2.4e-3) -	<b>1.15e+0 (2.0e-2)</b>
	8	5.11e+0 (1.5e-1) -	5.67e+0 (1.9e-1) -	3.54e+0 (5.9e-2) -	3.73e+0 (3.6e-2) -	3.74e+0 (2.1e-2) -	3.78e+0 (6.1e-3) -	<b>3.06e+0 (4.1e-2)</b>
	10	8.24e+0 (1.9e-1) -	7.53e+0 (2.6e-1) -	5.06e+0 (8.9e-2) -	5.40e+0 (8.1e-2) -	4.89e+0 (5.2e-2) -	5.00e+0 (1.1e-2) -	<b>4.11e+0 (2.9e-2)</b>
WFG6	3	2.62e-1 (1.8e-2) =	3.13e-1 (1.1e-2) -	4.63e-1 (3.0e-2) -	2.37e-1 (1.2e-2) +	<b>2.21e-1 (1.2e-2) +</b>	2.38e-1 (1.4e-2) +	2.71e-1 (2.1e-2)
	5	1.60e+0 (2.5e-2) -	2.87e+0 (1.9e-1) -	1.81e+0 (7.8e-2) -	1.49e+0 (5.2e-3) -	1.46e+0 (5.3e-3) -	1.49e+0 (1.5e-2) -	<b>1.21e+0 (2.3e-2)</b>
	8	5.60e+0 (1.4e-1) -	6.34e+0 (1.6e-1) -	3.95e+0 (5.5e-2) -	3.70e+0 (2.9e-2) -	3.77e+0 (1.7e-2) -	3.81e+0 (1.5e-2) -	<b>3.13e+0 (4.5e-2)</b>
	10	8.88e+0 (2.1e-1) -	8.14e+0 (2.9e-1) -	5.39e+0 (8.4e-2) +	5.30e+0 (9.6e-2) +	<b>5.04e+0 (2.8e-2) +</b>	5.04e+0 (2.6e-2) +	7.01e+0 (1.3e+0)
WFG7	3	2.94e-1 (4.4e-2) -	2.90e-1 (1.5e-3) -	3.63e-1 (2.4e-2) -	2.21e-1 (2.1e-3) +	<b>1.93e-1 (4.5e-4) +</b>	2.67e-1 (1.0e-2) -	2.51e-1 (9.5e-3)
	5	1.59e+0 (2.7e-2) -	2.93e+0 (1.5e-1) -	1.52e+0 (3.8e-2) -	1.49e+0 (1.6e-2) -	1.45e+0 (5.3e-3) -	1.51e+0 (1.3e-2) -	<b>1.19e+0 (1.9e-2)</b>
	8	5.36e+0 (2.0e-1) -	6.03e+0 (3.6e-1) -	3.69e+0 (6.2e-2) -	3.67e+0 (5.3e-2) -	3.75e+0 (1.9e-2) -	3.84e+0 (1.2e-2) -	<b>3.05e+0 (5.0e-2)</b>
	10	9.47e+0 (3.0e-1) -	7.64e+0 (5.0e-1) -	5.09e+0 (7.6e-2) -	5.24e+0 (2.1e-1) -	4.96e+0 (5.3e-2) -	5.09e+0 (3.0e-2) -	<b>4.10e+0 (5.5e-2)</b>
WFG8	3	2.87e-1 (4.5e-3) +	3.42e-1 (7.8e-3) -	4.25e-1 (1.5e-2) -	2.79e-1 (3.8e-3) +	<b>2.63e-1 (3.0e-3) +</b>	2.81e-1 (4.2e-3) +	3.11e-1 (1.0e-2)
	5	1.63e+0 (6.2e-2) -	2.28e+0 (1.3e-1) -	1.82e+0 (5.7e-2) -	1.55e+0 (5.7e-3) -	1.41e+0 (2.8e-2) -	1.55e+0 (1.4e-2) -	<b>1.27e+0 (1.9e-2)</b>
	8	5.33e+0 (2.9e-1) -	5.62e+0 (2.8e-1) -	4.10e+0 (5.0e-2) -	3.72e+0 (4.0e-2) -	3.80e+0 (1.9e-1) -	3.92e+0 (7.3e-2) -	<b>3.27e+0 (6.7e-2)</b>
	10	8.20e+0 (4.1e-1) -	7.76e+0 (2.9e-1) -	5.58e+0 (9.2e-2) =	5.35e+0 (1.2e-1) =	5.50e+0 (2.1e-1) =	<b>5.18e+0 (6.4e-2) =</b>	6.42e+0 (1.7e+0)
WFG9	3	2.48e-1 (2.4e-2) -	2.87e-1 (2.3e-3) -	2.67e-1 (1.2e-2) -	2.25e-1 (2.2e-2) -	<b>1.98e-1 (2.5e-3) +</b>	2.06e-1 (4.3e-3) +	2.24e-1 (7.5e-3)
	5	1.61e+0 (3.5e-2) -	2.54e+0 (1.4e-1) -	1.36e+0 (4.5e-2) -	1.49e+0 (1.3e-2) -	1.44e+0 (2.3e-2) -	1.46e+0 (8.2e-3) -	<b>1.15e+0 (2.9e-2)</b>
	8	5.19e+0 (1.9e-1) -	5.70e+0 (2.8e-1) -	3.55e+0 (7.2e-2) -	3.80e+0 (5.8e-2) -	3.73e+0 (6.5e-2) -	3.80e+0 (2.5e-2) -	<b>3.07e+0 (4.1e-2)</b>
	10	7.79e+0 (3.1e-1) -	7.53e+0 (3.4e-1) -	5.13e+0 (1.1e-1) -	5.33e+0 (1.9e-1) -	4.92e+0 (8.5e-2) -	5.02e+0 (2.2e-2) -	<b>4.11e+0 (4.4e-2)</b>
		+/-/=	5/27/4	4/31/1	4/30/2	10/24/2	18/15/3	15/17/4

Tabela 14 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família WFG com o método de inicialização  $k$ -layers para grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA	MOEADLNA
WFG1	3	2.45e-1 (2.1e-2) =	1.13e+0 (1.5e-1) -	3.69e-1 (1.3e-1) -	<b>1.56e-1 (8.2e-3) +</b>	2.25e-1 (3.4e-2) +	2.47e-1 (3.5e-2)
	5	6.39e-1 (3.2e-2) -	1.70e+0 (2.0e-1) -	9.52e-1 (1.1e-1) -	6.39e-1 (7.0e-2) =	<b>5.86e-1 (5.4e-2) +</b>	6.20e-1 (2.3e-2)
	8	1.38e+0 (9.5e-2) +	1.89e+0 (1.8e-1) -	1.68e+0 (8.1e-2) -	1.02e+0 (4.8e-2) +	<b>9.37e-1 (1.9e-2) +</b>	1.48e+0 (5.1e-2)
	10	1.56e+0 (1.1e-1) +	2.08e+0 (1.7e-1) -	1.79e+0 (5.4e-2) -	1.13e+0 (5.2e-2) +	<b>1.06e+0 (2.3e-2) +</b>	1.62e+0 (3.3e-2)
WFG2	3	<b>1.55e-1 (7.2e-3) +</b>	2.44e-1 (1.1e-2) -	2.41e-1 (2.0e-2) -	1.58e-1 (5.7e-3) +	1.70e-1 (7.9e-3) +	1.94e-1 (1.8e-2)
	5	4.81e-1 (1.3e-2) +	9.83e-1 (5.7e-2) -	6.16e-1 (3.5e-2) =	<b>4.47e-1 (1.1e-2) +</b>	5.01e-1 (7.0e-3) +	6.25e-1 (1.5e-2)
	8	1.05e+0 (2.8e-2) +	1.81e+0 (1.5e-1) -	1.08e+0 (4.0e-2) +	1.03e+0 (1.1e-1) +	<b>1.03e+0 (2.4e-2) +</b>	1.69e+0 (5.6e-2)
	10	1.20e+0 (4.2e-2) +	1.85e+0 (1.3e-1) =	1.23e+0 (3.3e-2) +	<b>1.10e+0 (5.2e-2) +</b>	1.16e+0 (2.4e-2) +	1.81e+0 (4.4e-2)
WFG3	3	1.71e-1 (8.7e-2) =	<b>8.25e-2 (5.0e-3) +</b>	2.68e-1 (3.5e-2) -	1.05e-1 (1.2e-2) +	1.99e-1 (1.7e-2) -	1.22e-1 (1.6e-2)
	5	4.40e-1 (8.3e-2) =	<b>1.44e-1 (9.3e-3) +</b>	3.99e-1 (3.8e-2) +	3.51e-1 (4.3e-2) +	4.90e-1 (2.9e-2) -	4.53e-1 (5.0e-2)
	8	1.84e+0 (1.4e-1) -	<b>5.20e-1 (2.9e-2) +</b>	8.00e-1 (1.4e-1) +	6.22e-1 (1.6e-1) +	1.27e+0 (6.7e-2) -	1.09e+0 (1.1e-1)
	10	2.58e+0 (1.6e-1) -	6.70e-1 (3.7e-2) +	1.04e+0 (2.1e-1) +	<b>3.00e-1 (9.2e-2) +</b>	1.79e+0 (6.1e-2) -	1.52e+0 (1.5e-1)
WFG4	3	<b>2.02e-1 (2.1e-3) +</b>	3.30e-1 (1.1e-2) -	2.85e-1 (1.6e-2) -	2.37e-1 (6.6e-3) -	2.04e-1 (3.3e-3) +	2.32e-1 (6.3e-3)
	5	<b>1.04e+0 (6.9e-3) +</b>	2.37e+0 (1.4e-1) -	1.14e+0 (2.4e-2) -	1.19e+0 (2.7e-2) -	1.47e+0 (4.2e-3) -	1.12e+0 (1.8e-2)
	8	<b>2.92e+0 (2.5e-2) +</b>	5.41e+0 (1.7e-1) -	3.02e+0 (5.4e-2) =	3.28e+0 (1.7e-1) -	3.75e+0 (1.8e-2) -	3.02e+0 (4.0e-2)
	10	<b>4.32e+0 (4.8e-2) =</b>	7.36e+0 (3.0e-1) -	4.57e+0 (1.7e-1) =	4.63e+0 (1.2e-1) =	4.97e+0 (3.8e-2) =	5.57e+0 (1.7e+0)
WFG5	3	2.16e-1 (2.1e-3) +	2.97e-1 (3.4e-3) -	3.40e-1 (2.3e-2) -	2.41e-1 (6.3e-3) -	<b>2.11e-1 (2.2e-3) +</b>	2.23e-1 (4.7e-3)
	5	<b>1.03e+0 (5.8e-3) +</b>	2.43e+0 (8.8e-2) -	1.23e+0 (2.5e-2) -	1.19e+0 (2.7e-2) -	1.46e+0 (4.6e-3) -	1.15e+0 (2.0e-2)
	8	<b>2.96e+0 (2.5e-2) +</b>	5.52e+0 (2.0e-1) -	3.12e+0 (4.7e-2) -	3.23e+0 (1.7e-1) -	3.72e+0 (2.1e-2) -	3.06e+0 (4.1e-2)
	10	4.25e+0 (2.9e-2) -	7.56e+0 (1.7e-1) -	4.50e+0 (5.8e-2) -	4.69e+0 (1.5e-1) -	4.94e+0 (2.7e-2) -	<b>4.11e+0 (2.9e-2)</b>
WFG6	3	2.43e-1 (2.9e-2) +	3.23e-1 (4.6e-2) -	5.60e-1 (3.0e-2) -	2.62e-1 (1.3e-2) =	<b>2.24e-1 (9.5e-3) +</b>	2.71e-1 (2.1e-2)
	5	<b>1.05e+0 (1.6e-2) +</b>	2.64e+0 (1.8e-1) -	1.53e+0 (6.0e-2) -	1.21e+0 (2.7e-2) =	1.50e+0 (1.3e-2) -	1.21e+0 (2.3e-2)
	8	<b>2.99e+0 (2.8e-2) +</b>	5.72e+0 (3.9e-1) -	4.03e+0 (4.6e-1) -	3.26e+0 (2.2e-1) -	3.78e+0 (2.7e-2) -	3.13e+0 (4.5e-2)
	10	<b>4.33e+0 (7.0e-2) +</b>	7.59e+0 (5.1e-1) =	6.07e+0 (6.0e-1) +	4.57e+0 (1.7e-1) +	5.09e+0 (7.0e-2) +	7.01e+0 (1.3e+0)
WFG7	3	2.02e-1 (3.5e-3) +	2.93e-1 (3.4e-3) -	5.14e-1 (3.2e-2) -	2.37e-1 (7.7e-3) +	<b>1.99e-1 (2.5e-3) +</b>	2.51e-1 (9.5e-3)
	5	<b>1.05e+0 (1.0e-2) +</b>	2.84e+0 (2.4e-1) -	1.42e+0 (4.9e-2) -	1.19e+0 (3.3e-2) =	1.47e+0 (7.3e-3) -	1.19e+0 (1.9e-2)
	8	3.13e+0 (2.1e-1) =	5.80e+0 (2.7e-1) -	3.55e+0 (3.7e-1) -	3.18e+0 (5.5e-2) -	3.72e+0 (1.9e-2) -	<b>3.05e+0 (5.0e-2)</b>
	10	5.08e+0 (2.6e-1) -	7.71e+0 (2.5e-1) -	5.52e+0 (7.2e-1) -	4.44e+0 (1.1e-1) -	5.02e+0 (3.6e-2) -	<b>4.10e+0 (5.5e-2)</b>
WFG8	3	2.85e-1 (1.7e-2) +	3.56e-1 (9.8e-3) -	6.08e-1 (2.4e-2) -	3.09e-1 (5.2e-3) =	<b>2.78e-1 (3.5e-3) +</b>	3.11e-1 (1.0e-2)
	5	<b>1.19e+0 (6.5e-2) +</b>	2.25e+0 (1.9e-1) -	1.68e+0 (1.3e-1) -	1.28e+0 (1.7e-2) -	1.57e+0 (1.3e-2) -	1.27e+0 (1.9e-2)
	8	3.27e+0 (7.2e-2) =	5.28e+0 (2.5e-1) -	4.73e+0 (3.2e-1) -	4.03e+0 (2.1e-1) -	4.11e+0 (4.3e-2) -	<b>3.27e+0 (6.7e-2)</b>
	10	<b>4.81e+0 (4.9e-1) =</b>	7.23e+0 (2.2e-1) =	6.83e+0 (3.7e-1) =	5.85e+0 (1.6e-1) =	5.55e+0 (9.4e-2) =	6.42e+0 (1.7e+0)
WFG9	3	2.33e-1 (4.9e-2) -	2.98e-1 (3.6e-2) -	3.17e-1 (4.8e-2) -	2.29e-1 (6.0e-3) -	<b>1.98e-1 (1.6e-3) +</b>	2.24e-1 (7.5e-3)
	5	<b>1.05e+0 (2.5e-2) +</b>	2.42e+0 (1.1e-1) -	1.17e+0 (6.3e-2) =	1.24e+0 (3.7e-2) -	1.46e+0 (7.2e-3) -	1.15e+0 (2.9e-2)
	8	3.12e+0 (1.2e-1) =	5.40e+0 (2.2e-1) -	3.98e+0 (7.6e-1) -	3.37e+0 (1.6e-1) -	3.73e+0 (1.9e-2) -	<b>3.07e+0 (4.1e-2)</b>
	10	4.56e+0 (3.7e-1) -	7.30e+0 (2.5e-1) -	5.70e+0 (9.1e-1) -	4.79e+0 (1.4e-1) -	4.92e+0 (3.9e-2) -	<b>4.11e+0 (4.4e-2)</b>
+/-/=		21/7/8	4/29/3	6/25/5	13/16/7	15/19/2	

Tabela 15 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família WFG com o método de inicialização Riesz S-Energy para grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEAD	MOEADDE	MOEADM2M	MOEADD	NSGAIII	MOEADLNA*	MOEADLNA
WFG1	3	2.33e-1 (1.5e-2) =	2.71e-1 (3.1e-3) -	1.56e+0 (1.3e-2) -	1.96e-1 (3.3e-2) +	<b>1.38e-1 (9.6e-3) +</b>	2.22e-1 (3.2e-2) =	2.36e-1 (3.6e-2)
	5	8.09e-1 (3.2e-2) +	1.23e+0 (7.6e-2) -	2.14e+0 (3.8e-2) -	6.42e-1 (6.9e-2) +	<b>4.45e-1 (2.3e-2) +</b>	8.73e-1 (6.7e-2) +	9.26e-1 (6.6e-2)
	8	1.72e+0 (8.2e-2) +	2.45e+0 (1.5e-1) -	2.68e+0 (2.6e-2) -	1.12e+0 (5.8e-2) +	<b>8.82e-1 (4.6e-2) +</b>	1.88e+0 (4.1e-2) =	1.87e+0 (3.6e-2)
	10	1.97e+0 (1.0e-1) +	2.41e+0 (1.2e-1) -	3.02e+0 (2.2e-2) -	1.40e+0 (1.1e-1) +	<b>1.21e+0 (7.7e-2) +</b>	2.13e+0 (1.0e-1) =	2.12e+0 (1.0e-1)
WFG2	3	2.25e-1 (2.0e-2) -	2.78e-1 (5.2e-3) -	2.26e-1 (1.1e-2) -	1.61e-1 (3.0e-3) =	<b>1.41e-1 (1.0e-3) +</b>	1.61e-1 (1.7e-2) +	1.81e-1 (2.5e-2)
	5	7.67e-1 (3.1e-2) +	1.26e+0 (6.3e-2) -	5.09e-1 (1.3e-2) +	5.38e-1 (1.0e-2) +	<b>4.19e-1 (2.1e-3) +</b>	7.94e-1 (7.0e-2) +	8.55e-1 (4.3e-2)
	8	1.97e+0 (2.7e-2) +	3.32e+0 (9.8e-1) -	1.08e+0 (3.0e-2) +	1.59e+0 (1.8e-2) +	<b>9.43e-1 (1.9e-1) +</b>	2.17e+0 (1.2e-1) =	2.20e+0 (1.2e-1)
	10	2.09e+0 (1.7e-2) +	6.70e+0 (2.5e+0) -	<b>1.23e+0 (3.1e-2) +</b>	1.79e+0 (2.5e-2) +	1.26e+0 (1.4e-1) +	2.28e+0 (1.0e-1) =	2.29e+0 (9.3e-2)
WFG3	3	1.28e-1 (1.0e-2) -	8.28e-2 (6.4e-4) +	3.45e-1 (3.7e-2) -	1.57e-1 (6.2e-3) -	8.93e-2 (7.2e-3) +	<b>7.46e-2 (4.0e-3) +</b>	1.16e-1 (1.3e-2)
	5	6.97e-1 (1.5e-1) -	1.56e+0 (2.6e-2) -	1.35e+0 (9.9e-2) -	6.37e-1 (3.1e-2) -	4.76e-1 (5.6e-2) +	<b>3.38e-1 (4.2e-2) +</b>	5.28e-1 (8.1e-2)
	8	5.39e+0 (1.8e-1) -	4.16e+0 (2.5e-2) -	2.91e+0 (3.2e-1) -	1.93e+0 (5.0e-2) =	<b>1.38e+0 (9.3e-2) +</b>	1.94e+0 (8.3e-1) =	2.45e+0 (1.4e+0)
	10	8.39e+0 (1.4e-1) -	5.44e+0 (2.9e-2) +	4.19e+0 (3.5e-1) +	2.98e+0 (9.9e-2) +	<b>1.96e+0 (1.8e-1) +</b>	6.11e+0 (1.0e+0) =	5.67e+0 (1.5e+0)
WFG4	3	2.25e-1 (4.4e-3) -	2.98e-1 (4.1e-3) -	2.92e-1 (1.1e-2) -	2.08e-1 (1.3e-3) +	<b>1.90e-1 (1.3e-4) +</b>	2.04e-1 (3.5e-3) +	2.21e-1 (5.1e-3)
	5	1.56e+0 (1.1e-1) -	2.47e+0 (1.1e-1) -	1.27e+0 (2.4e-2) -	1.15e+0 (5.9e-3) -	1.07e+0 (9.7e-2) -	1.03e+0 (7.7e-4) =	<b>1.03e+0 (5.7e-3)</b>
	8	5.88e+0 (8.6e-2) -	5.44e+0 (4.3e-1) -	3.52e+0 (6.8e-2) -	3.71e+0 (1.1e-1) -	2.92e+0 (3.3e-3) -	2.87e+0 (2.3e-3) -	<b>2.81e+0 (1.4e-2)</b>
	10	8.67e+0 (1.0e-1) -	9.04e+0 (5.3e-1) -	4.89e+0 (8.0e-2) -	5.91e+0 (1.1e-1) -	4.06e+0 (5.4e-3) -	4.00e+0 (6.0e-3) -	<b>3.95e+0 (7.7e-2)</b>
WFG5	3	2.20e-1 (2.3e-3) -	3.16e-1 (1.0e-3) -	2.60e-1 (6.8e-3) -	2.17e-1 (2.5e-3) =	<b>2.01e-1 (1.0e-4) +</b>	2.03e-1 (1.8e-3) +	2.19e-1 (5.0e-3)
	5	1.48e+0 (6.6e-2) -	2.66e+0 (1.0e-1) -	1.20e+0 (1.7e-2) -	1.13e+0 (6.2e-3) -	1.04e+0 (1.0e-3) -	1.02e+0 (1.4e-3) =	<b>1.02e+0 (5.2e-3)</b>
	8	5.64e+0 (9.6e-2) -	6.10e+0 (2.4e-1) -	3.30e+0 (4.0e-2) -	3.65e+0 (8.2e-2) -	2.90e+0 (3.9e-3) -	2.86e+0 (1.5e-3) -	<b>2.81e+0 (1.4e-2)</b>
	10	8.36e+0 (1.2e-1) -	9.53e+0 (5.3e-1) -	4.58e+0 (4.6e-2) -	6.08e+0 (9.9e-2) -	4.00e+0 (6.7e-3) -	3.98e+0 (2.4e-3) -	<b>3.91e+0 (1.7e-2)</b>
WFG6	3	2.55e-1 (1.3e-2) =	3.20e-1 (1.1e-2) -	4.39e-1 (3.3e-2) -	2.21e-1 (7.6e-3) +	<b>2.11e-1 (9.2e-3) +</b>	2.34e-1 (1.8e-2) +	2.57e-1 (1.5e-2)
	5	1.73e+0 (4.0e-2) -	2.63e+0 (6.0e-2) -	1.47e+0 (4.4e-2) -	1.14e+0 (9.0e-3) -	1.05e+0 (1.1e-3) -	<b>1.03e+0 (1.1e-3) +</b>	1.04e+0 (5.1e-3)
	8	6.06e+0 (8.3e-2) -	6.45e+0 (2.6e-1) -	3.59e+0 (5.0e-2) -	3.76e+0 (1.1e-1) -	2.91e+0 (2.8e-3) -	2.86e+0 (3.1e-3) -	<b>2.85e+0 (5.8e-2)</b>
	10	8.89e+0 (1.6e-1) -	9.52e+0 (5.4e-1) -	4.82e+0 (4.9e-2) +	6.13e+0 (1.3e-1) -	4.04e+0 (6.8e-3) +	<b>4.00e+0 (6.0e-2) +</b>	5.02e+0 (4.4e-1)
WFG7	3	2.63e-1 (1.8e-2) -	3.02e-1 (3.7e-3) -	3.30e-1 (1.7e-2) -	2.10e-1 (1.3e-3) +	<b>1.90e-1 (3.9e-4) +</b>	2.60e-1 (1.2e-2) -	2.40e-1 (7.8e-3)
	5	1.81e+0 (3.5e-2) -	2.59e+0 (7.1e-2) -	1.41e+0 (3.6e-2) -	1.16e+0 (7.4e-3) -	1.06e+0 (9.0e-4) -	1.04e+0 (2.4e-3) -	<b>1.03e+0 (4.4e-3)</b>
	8	6.13e+0 (5.5e-2) -	6.29e+0 (2.2e-1) -	3.48e+0 (6.1e-2) -	3.67e+0 (1.0e-1) -	2.92e+0 (2.9e-3) -	2.87e+0 (3.3e-3) -	<b>2.81e+0 (1.5e-2)</b>
	10	8.89e+0 (6.7e-2) -	8.17e+0 (4.9e-1) -	4.70e+0 (6.0e-2) -	5.81e+0 (1.2e-1) -	4.06e+0 (7.8e-3) -	<b>3.99e+0 (6.5e-3) +</b>	4.01e+0 (1.9e-1)
WFG8	3	2.85e-1 (1.4e-2) +	3.41e-1 (2.7e-3) -	3.90e-1 (1.8e-2) -	2.67e-1 (2.2e-3) +	<b>2.62e-1 (3.8e-3) +</b>	2.71e-1 (4.5e-3) +	3.03e-1 (7.7e-3)
	5	1.49e+0 (1.9e-1) -	2.81e+0 (9.4e-2) -	1.67e+0 (7.0e-2) -	1.14e+0 (1.1e-2) -	1.07e+0 (9.6e-3) +	<b>1.05e+0 (8.4e-3) +</b>	1.08e+0 (1.2e-2)
	8	5.67e+0 (1.7e-1) -	6.63e+0 (4.4e-2) -	3.85e+0 (5.2e-2) -	3.69e+0 (2.2e-1) -	2.95e+0 (2.0e-1) =	<b>2.88e+0 (2.5e-2) =</b>	3.12e+0 (4.8e-1)
	10	8.22e+0 (1.7e-1) -	1.03e+1 (1.3e-1) -	5.06e+0 (7.1e-2) =	5.95e+0 (2.3e-1) -	4.10e+0 (2.6e-1) +	<b>4.02e+0 (1.3e-1) +</b>	4.99e+0 (4.5e-1)
WFG9	3	2.45e-1 (3.1e-2) -	2.93e-1 (6.2e-3) -	2.44e-1 (7.5e-3) -	2.08e-1 (2.3e-3) +	<b>1.97e-1 (2.5e-2) +</b>	2.00e-1 (3.9e-3) +	2.16e-1 (6.9e-3)
	5	1.55e+0 (8.4e-2) -	2.67e+0 (6.9e-2) -	1.25e+0 (4.1e-2) -	1.13e+0 (8.7e-3) -	1.02e+0 (3.7e-3) -	<b>1.01e+0 (1.3e-3) =</b>	1.01e+0 (7.0e-3)
	8	5.82e+0 (6.7e-1) -	6.20e+0 (1.6e-1) -	3.34e+0 (4.9e-2) -	3.75e+0 (8.1e-2) -	2.83e+0 (7.9e-3) -	2.83e+0 (6.8e-3) -	<b>2.80e+0 (2.3e-2)</b>
	10	8.60e+0 (2.2e-1) -	9.52e+0 (4.2e-1) -	4.63e+0 (5.8e-2) -	6.07e+0 (1.2e-1) -	3.88e+0 (5.6e-2) =	3.90e+0 (1.1e-2) -	<b>3.87e+0 (1.9e-2)</b>
+/-/=		7/27/2	2/34/0	5/30/1	13/20/3	21/13/2	15/10/11	

Tabela 16 – Resultados de IGD (média e desvio padrão) para os algoritmos em 36 instâncias dos problemas da família WFG com o método de inicialização Riesz S-Energy para grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA	MOEADLNA
WFG1	3	3.26e-1 (5.3e-2) -	9.22e-1 (1.7e-1) -	3.74e-1 (1.4e-1) -	<b>1.51e-1 (7.3e-3) +</b>	2.15e-1 (2.4e-2) +	2.36e-1 (3.6e-2)
	5	2.27e+0 (5.0e-2) -	1.72e+0 (1.5e-1) -	9.51e-1 (1.4e-1) =	5.25e-1 (4.2e-2) +	<b>4.40e-1 (4.7e-2) +</b>	9.26e-1 (6.6e-2)
	8	2.97e+0 (8.5e-2) -	2.75e+0 (1.3e-1) -	1.91e+0 (1.8e-1) =	<b>8.67e-1 (4.6e-2) +</b>	8.98e-1 (3.9e-2) +	1.87e+0 (3.6e-2)
	10	3.34e+0 (5.3e-2) -	2.66e+0 (1.3e-1) -	2.03e+0 (1.6e-1) +	1.22e+0 (8.6e-2) +	<b>1.11e+0 (5.1e-2) +</b>	2.12e+0 (1.0e-1)
WFG2	3	1.65e-1 (6.5e-3) =	3.06e-1 (1.4e-2) -	2.33e-1 (2.4e-2) -	<b>1.50e-1 (3.1e-3) +</b>	1.63e-1 (6.1e-3) =	1.81e-1 (2.5e-2)
	5	9.13e-1 (9.9e-2) -	1.20e+0 (1.2e-1) -	7.14e-1 (3.4e-2) +	4.25e-1 (8.2e-3) +	<b>4.05e-1 (6.8e-3) +</b>	8.55e-1 (4.3e-2)
	8	2.50e+0 (1.1e+0) =	2.58e+0 (8.2e-1) -	1.31e+0 (9.3e-2) +	9.15e-1 (1.4e-1) +	<b>9.04e-1 (3.7e-2) +</b>	2.20e+0 (1.2e-1)
	10	3.74e+0 (1.7e+0) -	4.03e+0 (1.4e+0) -	1.74e+0 (3.1e-1) +	1.26e+0 (1.7e-1) +	<b>1.06e+0 (3.0e-2) +</b>	2.29e+0 (9.3e-2)
WFG3	3	1.41e-1 (6.1e-2) =	<b>9.20e-2 (4.3e-3) +</b>	2.51e-1 (3.8e-2) -	9.23e-2 (7.3e-3) +	1.91e-1 (1.0e-2) -	1.16e-1 (1.3e-2)
	5	9.65e-1 (1.4e-1) -	1.84e+0 (1.5e-1) -	4.58e-1 (5.7e-2) +	<b>4.48e-1 (7.7e-2) +</b>	5.05e-1 (2.9e-2) =	5.28e-1 (8.1e-2)
	8	1.54e+0 (2.9e-1) +	4.25e+0 (1.6e-1) -	<b>8.10e-1 (1.1e-1) +</b>	1.36e+0 (1.4e-1) +	2.31e+0 (4.2e-1) =	2.45e+0 (1.4e+0)
	10	1.90e+0 (3.4e-1) +	5.50e+0 (1.8e-1) +	<b>9.04e-1 (1.2e-1) +</b>	2.00e+0 (2.0e-1) +	3.67e+0 (6.0e-1) +	5.67e+0 (1.5e+0)
WFG4	3	2.09e-1 (6.1e-3) +	3.32e-1 (1.3e-2) -	2.75e-1 (1.6e-2) -	2.24e-1 (7.1e-3) =	<b>2.02e-1 (4.1e-3) +</b>	2.21e-1 (5.1e-3)
	5	1.40e+0 (7.0e-2) -	2.29e+0 (2.2e-1) -	1.15e+0 (2.1e-2) -	1.07e+0 (1.0e-2) -	1.05e+0 (9.8e-4) -	<b>1.03e+0 (5.7e-3)</b>
	8	4.61e+0 (3.9e-1) -	4.73e+0 (6.1e-1) -	3.02e+0 (1.7e-1) -	2.94e+0 (5.0e-2) -	2.92e+0 (1.7e-2) -	<b>2.81e+0 (1.4e-2)</b>
	10	7.34e+0 (7.0e-1) -	6.62e+0 (1.1e+0) -	5.81e+0 (5.2e-1) -	4.08e+0 (5.9e-2) -	4.22e+0 (3.3e-2) -	<b>3.95e+0 (7.7e-2)</b>
WFG5	3	2.14e-1 (3.2e-3) +	3.04e-1 (5.4e-3) -	3.38e-1 (2.2e-2) -	2.32e-1 (5.7e-3) -	<b>2.05e-1 (1.2e-3) +</b>	2.19e-1 (5.0e-3)
	5	1.43e+0 (4.7e-2) -	2.71e+0 (1.5e-1) -	1.19e+0 (5.0e-2) -	1.06e+0 (1.0e-2) -	1.04e+0 (1.4e-3) -	<b>1.02e+0 (5.2e-3)</b>
	8	4.14e+0 (3.7e-1) -	5.73e+0 (2.7e-1) -	3.16e+0 (6.4e-2) -	2.90e+0 (3.8e-3) -	2.92e+0 (1.5e-2) -	<b>2.81e+0 (1.4e-2)</b>
	10	6.27e+0 (7.0e-1) -	8.77e+0 (5.2e-1) -	4.74e+0 (2.3e-1) -	4.00e+0 (7.3e-3) -	4.25e+0 (3.3e-2) -	<b>3.91e+0 (1.7e-2)</b>
WFG6	3	2.42e-1 (2.2e-2) +	3.73e-1 (5.2e-2) -	5.61e-1 (3.1e-2) -	2.51e-1 (1.2e-2) =	<b>2.23e-1 (1.1e-2) +</b>	2.57e-1 (1.5e-2)
	5	1.63e+0 (8.2e-2) -	2.77e+0 (2.4e-1) -	1.48e+0 (3.8e-2) -	1.08e+0 (1.1e-2) -	1.05e+0 (1.7e-3) -	<b>1.04e+0 (5.1e-3)</b>
	8	4.99e+0 (9.9e-1) -	5.92e+0 (4.6e-1) -	3.64e+0 (1.7e-1) -	2.91e+0 (2.3e-3) -	2.91e+0 (2.2e-2) -	<b>2.85e+0 (5.8e-2)</b>
	10	7.27e+0 (1.2e+0) -	9.04e+0 (7.5e-1) -	5.59e+0 (3.0e-1) -	<b>4.04e+0 (5.3e-3) +</b>	4.27e+0 (1.0e-1) +	5.02e+0 (4.4e-1)
WFG7	3	2.02e-1 (5.2e-3) +	3.03e-1 (3.4e-3) -	5.06e-1 (3.4e-2) -	2.26e-1 (7.4e-3) +	<b>1.98e-1 (2.5e-3) +</b>	2.40e-1 (7.8e-3)
	5	1.52e+0 (8.6e-2) -	2.76e+0 (2.0e-1) -	1.40e+0 (4.2e-2) -	1.07e+0 (8.2e-3) -	1.05e+0 (7.2e-4) -	<b>1.03e+0 (4.4e-3)</b>
	8	5.02e+0 (7.4e-1) -	5.85e+0 (4.6e-1) -	3.30e+0 (1.3e-1) -	2.92e+0 (2.8e-3) -	2.95e+0 (2.4e-2) -	<b>2.81e+0 (1.5e-2)</b>
	10	7.82e+0 (1.3e+0) -	8.54e+0 (6.5e-1) -	4.69e+0 (4.3e-1) -	4.10e+0 (1.7e-1) -	4.32e+0 (4.3e-2) -	<b>4.01e+0 (1.9e-1)</b>
WFG8	3	2.85e-1 (4.9e-3) +	3.57e-1 (6.2e-3) -	6.01e-1 (2.8e-2) -	2.98e-1 (5.1e-3) +	<b>2.80e-1 (7.3e-3) +</b>	3.03e-1 (7.7e-3)
	5	1.79e+0 (9.2e-2) -	3.09e+0 (1.6e-1) -	1.56e+0 (5.2e-2) -	1.14e+0 (1.6e-2) -	<b>1.07e+0 (2.3e-3) +</b>	1.08e+0 (1.2e-2)
	8	5.24e+0 (7.0e-1) -	6.68e+0 (3.2e-1) -	4.10e+0 (2.2e-1) -	<b>2.93e+0 (1.6e-1) =</b>	2.94e+0 (4.9e-2) +	3.12e+0 (4.8e-1)
	10	7.68e+0 (1.3e+0) -	9.84e+0 (5.6e-1) -	6.20e+0 (2.4e-1) -	<b>4.05e+0 (2.2e-1) +</b>	4.28e+0 (1.1e-1) +	4.99e+0 (4.5e-1)
WFG9	3	2.38e-1 (5.5e-2) =	3.01e-1 (2.9e-2) -	3.08e-1 (5.0e-2) -	2.21e-1 (2.8e-2) =	<b>1.97e-1 (5.4e-3) +</b>	2.16e-1 (6.9e-3)
	5	1.70e+0 (1.1e-1) -	2.63e+0 (1.7e-1) -	1.18e+0 (3.7e-2) -	1.03e+0 (9.1e-3) -	1.03e+0 (1.9e-3) -	<b>1.01e+0 (7.0e-3)</b>
	8	5.05e+0 (5.9e-1) -	5.65e+0 (2.4e-1) -	3.39e+0 (1.6e-1) -	2.83e+0 (1.2e-2) -	2.89e+0 (1.7e-2) -	<b>2.80e+0 (2.3e-2)</b>
	10	7.15e+0 (6.5e-1) -	8.51e+0 (6.3e-1) -	5.18e+0 (3.1e-1) -	3.90e+0 (7.9e-2) -	4.22e+0 (4.9e-2) -	<b>3.87e+0 (1.9e-2)</b>
+/-/=		7/25/4	2/34/0	7/27/2	16/16/4	18/15/3	

Tabela 17 – Resultados de IGD (média e desvio padrão) para os algoritmos em 8 instâncias dos problemas da família GPD com o método de inicialização  $k$ -layers para o grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEAD	MOEADDE	MOEADM2M	MOEADD	NSGAIII	MOEADLNA*	MOEADLNA
GPD01	3	4.72e-2 (9.9e-5) -	6.23e-2 (3.9e-4) -	8.32e-2 (7.9e-3) -	4.51e-2 (6.3e-4) -	1.40e-1 (4.6e-2) -	2.99e-1 (1.6e-5) -	3.78e-2 (3.7e-4)
	5	2.39e-1 (9.0e-4) -	2.99e-1 (2.6e-2) -	4.39e-1 (3.8e-2) -	2.37e-1 (4.2e-3) -	2.67e-1 (3.7e-2) -	4.47e-1 (2.4e-4) -	1.84e-1 (4.7e-3)
	8	4.18e-1 (5.1e-4) -	5.22e-1 (2.2e-2) -	7.11e-1 (3.8e-2) -	4.17e-1 (9.8e-4) -	4.03e-1 (2.2e-2) -	5.45e-1 (1.9e-4) -	3.49e-1 (4.7e-3)
	10	4.46e-1 (3.7e-4) -	6.10e-1 (2.3e-2) -	7.70e-1 (3.2e-2) -	4.46e-1 (8.2e-4) -	4.45e-1 (1.4e-2) -	5.94e-1 (2.5e-4) -	3.84e-1 (4.0e-3)
GPD02	3	4.45e-2 (2.7e-5) -	4.85e-2 (3.3e-4) -	1.85e-1 (2.0e-2) -	4.14e-2 (2.1e-4) -	3.81e-2 (4.1e-4) =	4.45e-2 (2.0e-5) -	3.82e-2 (5.7e-4)
	5	2.71e-1 (2.3e-4) -	4.81e-1 (2.2e-2) -	7.43e-1 (4.9e-2) -	2.52e-1 (4.5e-3) =	2.33e-1 (4.0e-3) +	2.71e-1 (3.4e-4) -	2.54e-1 (4.5e-3)
	8	4.73e-1 (4.1e-4) -	7.09e-1 (2.6e-2) -	1.01e+0 (5.3e-2) -	4.63e-1 (2.0e-3) -	4.56e-1 (1.6e-2) -	4.73e-1 (4.7e-4) -	4.23e-1 (5.0e-3)
	10	5.89e-1 (5.2e-4) -	8.62e-1 (3.9e-2) -	1.14e+0 (5.6e-2) -	5.84e-1 (2.3e-3) -	5.98e-1 (4.2e-2) -	5.88e-1 (3.3e-4) -	5.20e-1 (4.2e-3)
+/-/=		0/8/0	0/8/0	0/8/0	0/7/1	1/6/1	0/8/0	

Tabela 18 – Resultados de IGD (média e desvio padrão) para os algoritmos em 8 instâncias dos problemas da família GPD com o método de inicialização  $k$ -layers para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA	MOEADLNA
GPD01	3	4.21e-2 (1.0e-3) -	6.24e-2 (9.0e-4) -	9.05e-2 (7.7e-3) -	1.62e-1 (3.6e-2) -	4.43e-2 (1.4e-4) -	3.78e-2 (3.7e-4)
	5	2.00e-1 (1.5e-2) -	3.93e-1 (1.9e-2) -	2.67e-1 (2.6e-2) -	2.79e-1 (3.5e-2) -	2.40e-1 (1.8e-3) -	1.84e-1 (4.7e-3)
	8	3.73e-1 (1.1e-2) -	6.46e-1 (1.4e-2) -	4.24e-1 (2.2e-2) -	4.02e-1 (1.9e-2) -	4.21e-1 (6.0e-4) -	3.49e-1 (4.7e-3)
	10	4.26e-1 (1.1e-2) -	6.84e-1 (1.4e-2) -	5.20e-1 (2.5e-2) -	4.58e-1 (2.2e-2) -	4.50e-1 (7.0e-4) -	3.84e-1 (4.0e-3)
GPD02	3	3.96e-2 (1.4e-3) -	4.78e-2 (9.6e-4) -	8.36e-2 (5.1e-3) -	3.83e-2 (2.8e-4) =	4.49e-2 (8.6e-4) -	3.82e-2 (5.7e-4)
	5	2.02e-1 (3.6e-3) +	5.60e-1 (3.5e-2) -	2.91e-1 (2.3e-2) -	2.17e-1 (5.1e-3) +	2.71e-1 (1.1e-3) -	2.54e-1 (4.5e-3)
	8	4.59e-1 (1.0e-1) =	8.25e-1 (4.0e-2) -	6.36e-1 (6.5e-2) -	4.49e-1 (7.5e-2) -	4.63e-1 (8.1e-3) -	4.23e-1 (5.0e-3)
	10	7.04e-1 (1.2e-1) -	9.69e-1 (3.6e-2) -	8.24e-1 (6.8e-2) -	6.40e-1 (7.7e-2) -	5.78e-1 (6.3e-3) -	5.20e-1 (4.2e-3)
+/-/=		1/6/1	0/8/0	0/8/0	1/6/1	0/8/0	

Tabela 19 – Resultados de IGD (média e desvio padrão) para os algoritmos em 8 instâncias dos problemas da família GPD com o método de inicialização Riesz S-Energy para o grupo 1. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEAD	MOEADDE	MOEADM2M	MOEADD	NSGAIII	MOEADLNA*	MOEADLNA
GPD01	3	2.99e-1 (1.9e-5) -	3.03e-1 (2.0e-4) -	3.29e-1 (8.5e-3) -	2.99e-1 (5.5e-5) -	3.06e-1 (6.5e-3) -	2.99e-1 (9.7e-6) -	<b>2.98e-1 (3.3e-4)</b>
	5	4.22e-1 (7.0e-5) -	4.92e-1 (7.1e-3) -	6.38e-1 (3.1e-2) -	4.22e-1 (2.4e-4) -	4.57e-1 (1.2e-2) -	<b>4.21e-1 (7.8e-5) +</b>	4.21e-1 (5.2e-4)
	8	5.02e-1 (1.8e-4) +	6.42e-1 (9.0e-3) -	7.68e-1 (2.2e-2) -	5.03e-1 (2.7e-4) +	5.35e-1 (1.1e-2) -	<b>5.02e-1 (2.0e-4) +</b>	5.04e-1 (1.2e-3)
	10	5.62e-1 (3.8e-4) =	7.02e-1 (1.2e-2) -	8.14e-1 (2.9e-2) -	5.63e-1 (3.2e-4) -	5.90e-1 (8.6e-3) -	<b>5.62e-1 (3.3e-4) +</b>	5.63e-1 (8.2e-4)
GPD02	3	4.00e-2 (9.4e-6) -	5.58e-2 (4.6e-4) -	1.43e-1 (1.8e-2) -	3.88e-2 (2.2e-4) -	3.60e-2 (3.7e-4) -	4.00e-2 (1.2e-5) -	<b>3.49e-2 (8.9e-4)</b>
	5	1.91e-1 (1.2e-4) +	4.45e-1 (1.5e-2) -	6.85e-1 (6.3e-2) -	1.91e-1 (3.8e-4) +	<b>1.89e-1 (3.0e-4) +</b>	1.91e-1 (1.0e-4) +	1.95e-1 (8.5e-4)
	8	3.63e-1 (7.8e-5) +	6.38e-1 (9.2e-3) -	9.27e-1 (6.1e-2) -	<b>3.63e-1 (7.2e-5) +</b>	3.66e-1 (2.4e-2) +	3.63e-1 (7.4e-5) +	3.68e-1 (1.2e-3)
	10	4.86e-1 (1.8e-4) -	7.35e-1 (2.4e-2) -	1.00e+0 (4.6e-2) -	4.86e-1 (1.6e-4) =	5.02e-1 (4.4e-2) -	4.86e-1 (1.4e-4) -	<b>4.85e-1 (7.5e-4)</b>
+/-/=		3/4/1	0/8/0	0/8/0	3/4/1	2/6/0	5/3/0	

Tabela 20 – Resultados de IGD (média e desvio padrão) para os algoritmos em 8 instâncias dos problemas da família GPD com o método de inicialização Riesz S-Energy para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA	MOEADLNA
GPD01	3	3.00e-1 (2.6e-4) -	3.09e-1 (9.7e-4) -	3.10e-1 (1.2e-3) -	3.32e-1 (1.8e-2) -	2.99e-1 (3.6e-5) -	<b>2.98e-1 (3.3e-4)</b>
	5	6.46e-1 (3.7e-2) -	6.11e-1 (2.3e-2) -	4.46e-1 (2.1e-3) -	4.59e-1 (1.0e-2) -	4.21e-1 (1.8e-4) =	<b>4.21e-1 (5.2e-4)</b>
	8	9.44e-1 (9.9e-2) -	1.09e+0 (9.1e-2) -	5.61e-1 (5.7e-3) -	5.36e-1 (1.0e-2) -	<b>5.01e-1 (3.1e-4) +</b>	5.04e-1 (1.2e-3)
	10	1.06e+0 (1.0e-1) -	1.16e+0 (1.0e-1) -	6.43e-1 (8.5e-3) -	5.88e-1 (1.1e-2) -	<b>5.62e-1 (3.0e-4) +</b>	5.63e-1 (8.2e-4)
GPD02	3	3.93e-2 (1.5e-3) -	5.94e-2 (2.7e-3) -	8.47e-2 (6.4e-3) -	<b>3.34e-2 (4.4e-4) +</b>	4.03e-2 (7.4e-4) -	3.49e-2 (8.9e-4)
	5	9.11e-1 (8.2e-2) -	7.37e-1 (4.8e-2) -	2.92e-1 (6.6e-3) -	<b>1.89e-1 (4.4e-4) +</b>	1.91e-1 (2.5e-3) +	1.95e-1 (8.5e-4)
	8	1.57e+0 (1.6e-1) -	1.47e+0 (2.0e-1) -	4.76e-1 (8.2e-3) -	<b>3.61e-1 (2.8e-4) +</b>	3.63e-1 (1.2e-4) +	3.68e-1 (1.2e-3)
	10	1.68e+0 (1.7e-1) -	1.61e+0 (1.8e-1) -	6.11e-1 (1.3e-2) -	4.96e-1 (4.1e-2) -	4.86e-1 (1.9e-4) -	<b>4.85e-1 (7.5e-4)</b>
+/-/=		0/8/0	0/8/0	0/8/0	3/5/0	4/3/1	



Tabela 21 – Resultados de IGD (média e desvio padrão) para os algoritmos em 48 instâncias dos problemas da família MaF com o método de inicialização *k-layers* para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (−) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA	MOEADLNA
MaF1	3	4.63e-2 (1.2e-3) -	5.76e-2 (1.2e-4) -	1.48e-1 (1.6e-2) -	4.41e-2 (1.1e-3) -	6.43e-2 (2.7e-4) -	4.00e-2 (4.3e-4)
	5	2.34e-1 (4.7e-2) -	1.86e-1 (2.2e-3) -	2.21e-1 (2.6e-2) -	1.39e-1 (5.1e-3) -	4.42e-1 (8.6e-2) -	1.18e-1 (1.8e-3)
	8	4.36e-1 (5.8e-2) -	3.37e-1 (8.8e-3) -	2.64e-1 (1.1e-2) -	2.74e-1 (1.2e-2) -	6.07e-1 (9.5e-2) -	2.33e-1 (2.0e-2)
	10	3.98e-1 (7.4e-2) =	3.75e-1 (2.0e-2) =	2.89e-1 (1.4e-2) +	2.78e-1 (1.0e-2) +	6.44e-1 (6.2e-2) -	3.64e-1 (6.0e-2)
MaF2	3	2.83e-2 (1.6e-4) +	4.00e-2 (6.7e-4) -	6.32e-2 (3.1e-3) -	3.15e-2 (1.1e-3) +	3.62e-2 (9.9e-4) -	3.27e-2 (1.3e-3)
	5	1.05e-1 (2.4e-3) -	1.47e-1 (4.1e-3) -	1.08e-1 (2.5e-3) -	1.05e-1 (5.2e-3) -	1.48e-1 (2.7e-2) -	9.09e-2 (1.8e-3)
	8	1.85e-1 (9.2e-3) =	2.56e-1 (1.2e-2) -	2.38e-1 (1.7e-2) -	3.97e-1 (3.1e-2) -	2.94e-1 (3.9e-2) -	1.86e-1 (1.0e-2)
	10	1.97e-1 (7.5e-3) +	2.65e-1 (3.0e-2) =	2.75e-1 (1.3e-2) +	3.55e-1 (2.8e-2) -	3.52e-1 (6.7e-2) -	3.26e-1 (1.8e-1)
MaF3	3	5.46e-2 (5.4e-3) =	2.72e-1 (5.7e-1) -	1.09e-1 (4.8e-2) -	5.35e-2 (5.1e-3) +	9.77e-2 (1.9e-1) =	5.62e-2 (4.7e-3)
	5	7.98e-2 (5.0e-3) +	2.16e-1 (2.2e-2) -	1.53e-1 (1.6e-2) -	1.43e-1 (2.9e-1) +	8.05e-1 (1.7e+0) =	1.45e-1 (3.7e-2)
	8	1.03e-1 (7.0e-3) +	2.13e-1 (9.4e-3) +	1.81e-1 (2.0e-2) +	7.18e+1 (1.0e+2) -	6.55e-1 (2.9e+0) -	3.60e-1 (1.4e-1)
	10	1.10e-1 (8.1e-3) +	1.74e-1 (7.1e-3) +	1.62e-1 (1.7e-2) +	2.21e+3 (3.5e+3) -	4.51e-1 (7.3e-1) -	4.14e-1 (1.3e-1)
MaF4	3	7.39e-1 (2.7e-1) -	3.93e+0 (9.8e+0) -	2.94e-1 (3.0e-2) +	3.47e-1 (3.0e-2) -	8.62e-1 (7.8e-1) -	3.12e-1 (2.3e-2)
	5	4.39e+0 (1.5e+0) -	3.58e+0 (1.5e+0) =	2.11e+0 (1.6e-1) +	3.30e+0 (3.3e-1) -	5.11e+0 (4.7e-1) -	3.11e+0 (2.5e-1)
	8	3.04e+1 (8.3e+0) -	3.01e+1 (7.6e+0) -	1.60e+1 (6.8e-1) +	6.01e+1 (8.2e+1) -	4.51e+1 (7.1e+0) -	2.35e+1 (2.5e+0)
	10	1.07e+2 (2.7e+1) -	1.12e+2 (5.6e+0) -	5.86e+1 (3.3e+0) +	9.76e+1 (1.9e+1) =	1.73e+2 (3.5e+1) -	8.90e+1 (8.0e+0)
MaF5	3	5.97e-1 (9.7e-1) -	5.96e-1 (5.2e-1) -	1.08e+0 (1.0e+0) -	4.46e-1 (4.8e-1) -	2.21e-1 (6.2e-4) +	3.55e-1 (4.5e-1)
	5	3.54e+0 (1.7e+0) -	1.07e+1 (9.5e-1) -	2.87e+0 (1.5e+0) =	2.21e+0 (2.8e-1) +	2.39e+0 (1.3e-1) +	2.54e+0 (1.3e+0)
	8	2.88e+1 (8.7e+0) +	8.22e+1 (7.6e+0) -	2.68e+1 (9.2e+0) +	3.36e+1 (1.6e+1) +	3.06e+1 (4.2e+0) +	4.80e+1 (2.3e+1)
	10	1.10e+2 (3.2e+1) +	2.82e+2 (3.3e+1) -	9.61e+1 (2.6e+1) +	1.17e+2 (4.5e+1) +	1.45e+2 (3.4e+1) +	2.17e+2 (6.8e+1)
MaF6	3	7.27e-3 (4.4e-4) -	1.39e-2 (1.2e-4) -	6.07e-2 (1.7e-2) -	1.59e-2 (2.8e-3) -	4.54e-2 (1.4e-2) -	6.34e-3 (3.3e-4)
	5	3.05e-2 (4.0e-2) -	6.55e-3 (1.1e-4) +	5.59e-2 (1.1e-2) -	1.39e-2 (2.7e-3) -	9.99e-2 (5.7e-2) -	7.31e-3 (7.8e-4)
	8	3.85e-2 (4.5e-2) -	7.63e-3 (1.7e-4) =	6.05e-2 (1.2e-2) -	3.26e-1 (1.8e-1) -	8.59e-2 (4.7e-2) -	7.60e-3 (1.6e-3)
	10	4.36e-2 (4.7e-2) -	8.36e-3 (1.3e-4) +	5.64e-2 (1.1e-2) -	5.52e-1 (2.9e-1) -	6.51e-2 (1.9e-2) -	8.48e-3 (1.9e-3)
MaF7	3	1.08e-1 (8.5e-2) -	3.34e-1 (2.2e-1) -	1.25e-1 (4.6e-2) -	7.02e-2 (2.3e-3) -	9.21e-2 (1.6e-3) -	6.16e-2 (2.4e-3)
	5	4.36e-1 (8.1e-2) -	1.22e+0 (1.0e-1) -	4.58e-1 (2.8e-2) -	3.72e-1 (1.7e-2) -	4.77e-1 (1.3e-2) -	3.11e-1 (1.1e-2)
	8	1.06e+0 (5.4e-2) -	1.64e+0 (4.9e-1) -	1.16e+0 (9.6e-2) -	9.18e-1 (6.9e-2) -	9.11e-1 (5.7e-2) -	6.16e-1 (7.2e-3)
	10	1.48e+0 (2.1e-1) -	1.90e+0 (4.5e-1) -	1.60e+0 (8.0e-2) -	1.34e+0 (1.4e-1) -	1.16e+0 (5.2e-2) -	7.44e-1 (7.2e-3)

MaF8	3	1.01e-1 (3.4e-2) =	8.08e-2 (1.2e-3) +	8.67e-2 (5.7e-3) +	9.17e-2 (6.4e-3) =	1.30e-1 (1.2e-2) -	1.13e-1 (8.5e-2)
	5	4.19e-1 (6.5e-2) -	1.10e-1 (9.1e-4) +	1.17e-1 (4.8e-3) +	1.78e-1 (3.1e-2) =	4.35e-1 (7.0e-2) -	1.83e-1 (4.5e-2)
	8	6.11e-1 (6.7e-2) -	1.50e-1 (1.5e-3) +	1.35e-1 (3.8e-3) +	2.08e-1 (2.4e-2) +	8.28e-1 (1.4e-1) -	3.77e-1 (4.8e-2)
	10	6.58e-1 (8.3e-2) -	1.64e-1 (1.5e-3) +	1.40e-1 (3.0e-3) +	1.89e-1 (1.6e-2) +	1.05e+0 (1.8e-1) -	5.29e-1 (3.7e-2)
MaF9	3	7.57e-2 (8.1e-3) =	7.14e-2 (5.9e-4) =	2.37e-1 (1.2e-1) -	6.39e-2 (3.8e-3) +	1.16e-1 (1.0e-1) =	8.55e-2 (2.8e-2)
	5	2.78e-1 (6.2e-2) -	9.48e-2 (4.2e-4) +	2.17e-1 (4.0e-2) -	5.35e-1 (1.7e-1) -	3.57e-1 (6.2e-2) -	1.29e-1 (1.9e-2)
	8	6.24e-1 (9.2e-2) -	1.41e-1 (3.7e-3) +	7.82e-1 (2.4e-1) -	3.46e-1 (1.3e-1) -	6.56e-1 (1.2e-1) -	1.83e-1 (4.2e-2)
	10	1.33e+0 (1.4e-1) -	1.54e-1 (2.4e-3) +	1.21e+0 (8.6e-1) -	2.63e-1 (5.7e-2) -	7.71e-1 (1.7e-1) -	2.30e-1 (2.1e-2)
MaF10	3	2.57e-1 (1.9e-2) +	1.29e+0 (1.4e-1) -	4.02e-1 (1.6e-1) =	1.92e-1 (2.6e-2) +	3.14e-1 (4.0e-2) +	3.67e-1 (6.5e-2)
	5	6.80e-1 (4.8e-2) -	1.84e+0 (1.8e-1) -	9.87e-1 (1.0e-1) -	8.12e-1 (1.1e-1) -	6.82e-1 (7.0e-2) -	6.51e-1 (3.3e-2)
	8	1.42e+0 (1.0e-1) +	2.00e+0 (1.6e-1) -	1.66e+0 (6.8e-2) -	1.12e+0 (9.0e-2) +	1.01e+0 (4.2e-2) +	1.48e+0 (3.3e-2)
	10	1.60e+0 (1.0e-1) =	2.17e+0 (1.7e-1) -	1.79e+0 (5.8e-2) -	1.21e+0 (7.9e-2) +	1.10e+0 (3.2e-2) +	1.62e+0 (1.8e-2)
MaF11	3	1.63e-1 (7.8e-3) +	2.49e-1 (1.4e-2) -	2.52e-1 (2.6e-2) -	1.57e-1 (4.5e-3) +	1.73e-1 (5.9e-3) +	1.95e-1 (1.8e-2)
	5	5.05e-1 (1.9e-2) +	9.82e-1 (8.9e-2) -	6.16e-1 (2.7e-2) =	4.47e-1 (1.2e-2) +	4.99e-1 (5.3e-3) +	6.26e-1 (1.3e-2)
	8	1.05e+0 (2.1e-2) +	1.79e+0 (1.7e-1) -	1.09e+0 (3.5e-2) +	1.00e+0 (8.8e-2) +	1.04e+0 (2.4e-2) +	1.65e+0 (4.4e-2)
	10	1.22e+0 (3.4e-2) +	1.85e+0 (1.2e-1) =	1.24e+0 (3.8e-2) +	1.10e+0 (5.1e-2) +	1.17e+0 (2.6e-2) +	1.80e+0 (3.8e-2)
MaF12	3	2.34e-1 (5.2e-2) -	2.87e-1 (2.9e-3) -	3.11e-1 (4.8e-2) -	2.30e-1 (6.5e-3) =	2.00e-1 (2.4e-3) +	2.30e-1 (9.8e-3)
	5	1.10e+0 (2.1e-2) +	2.37e+0 (1.3e-1) -	1.20e+0 (1.4e-1) =	1.24e+0 (3.0e-2) -	1.46e+0 (6.5e-3) -	1.18e+0 (2.3e-2)
	8	3.10e+0 (1.6e-1) +	5.42e+0 (2.7e-1) -	3.82e+0 (5.6e-1) -	3.37e+0 (1.5e-1) -	3.73e+0 (2.7e-2) -	3.13e+0 (4.5e-2)
	10	4.53e+0 (2.1e-1) -	7.23e+0 (2.4e-1) -	5.88e+0 (1.0e+0) -	4.78e+0 (1.7e-1) -	4.93e+0 (4.2e-2) -	4.14e+0 (4.3e-2)
+/-/=		15/27/6	11/31/6	16/28/4	17/27/4	12/33/3	

Tabela 22 – Resultados de IGD (média e desvio padrão) para os algoritmos em 48 instâncias dos problemas da família MaF com o método de inicialização Riesz S-Energy para o grupo 2. A melhor média para cada caso é destacada em azul. A última linha mostra o número de vitórias (+), perdas (-) e empates (=) para cada algoritmo contra MOEA/D-LNA.

Problema	M	MOEADAWA	MOEADDRA	EAGMOEAD	ANSGAIII	RVEA	MOEADLNA
MaF1	3	4.87e-2 (1.4e-3) -	5.98e-2 (3.1e-4) -	1.51e-1 (1.7e-2) -	4.03e-2 (1.3e-3) -	7.06e-2 (8.7e-4) -	<b>3.74e-2 (3.2e-4)</b>
	5	3.85e-1 (7.7e-2) -	1.74e-1 (2.8e-3) -	2.27e-1 (1.7e-2) -	1.54e-1 (7.9e-3) -	2.86e-1 (1.3e-2) -	<b>1.20e-1 (1.2e-3)</b>
	8	7.07e-1 (1.0e-1) -	2.44e-1 (3.0e-3) -	2.74e-1 (8.6e-3) -	<b>2.26e-1 (5.2e-3) +</b>	5.47e-1 (8.6e-2) -	2.27e-1 (3.1e-2)
	10	7.00e-1 (1.1e-1) -	2.55e-1 (3.1e-3) +	2.93e-1 (1.1e-2) =	<b>2.41e-1 (2.9e-3) +</b>	6.03e-1 (8.0e-2) -	3.06e-1 (5.5e-2)
MaF2	3	<b>2.88e-2 (3.1e-4) +</b>	3.89e-2 (5.9e-4) -	6.18e-2 (2.7e-3) -	2.93e-2 (6.3e-4) +	3.76e-2 (1.2e-3) -	3.11e-2 (1.1e-3)
	5	1.33e-1 (5.0e-3) -	1.50e-1 (3.1e-3) -	1.12e-1 (2.8e-3) -	1.09e-1 (2.0e-3) -	1.25e-1 (1.0e-3) -	<b>1.01e-1 (1.9e-3)</b>
	8	2.62e-1 (4.6e-2) -	3.37e-1 (2.5e-2) -	<b>1.68e-1 (3.6e-3) +</b>	1.69e-1 (5.7e-3) +	1.99e-1 (5.7e-3) +	2.26e-1 (4.2e-2)
	10	2.98e-1 (5.7e-2) +	3.80e-1 (3.5e-2) +	<b>1.96e-1 (1.1e-2) +</b>	2.04e-1 (2.0e-2) +	2.69e-1 (1.1e-1) +	5.27e-1 (5.5e-2)
MaF3	3	2.08e+0 (2.9e+0) -	8.96e+1 (4.9e+2) -	7.98e-2 (4.2e-2) =	1.20e-1 (4.0e-1) -	1.13e+0 (4.6e+0) -	<b>5.09e-2 (2.8e-3)</b>
	5	6.44e+4 (3.8e+4) -	7.75e+3 (1.0e+4) -	1.44e-1 (1.1e-2) =	1.56e-1 (3.1e-1) -	4.35e-1 (1.5e+0) =	<b>1.42e-1 (3.5e-2)</b>
	8	4.46e+9 (1.0e+10) -	3.10e+4 (1.5e+4) -	1.42e+3 (1.7e+3) -	2.52e+2 (3.0e+2) -	<b>1.27e-1 (4.4e-2) +</b>	4.08e-1 (1.2e-1)
	10	3.59e+9 (6.4e+9) -	5.02e+4 (1.2e+4) -	9.31e+4 (7.2e+4) -	1.19e+4 (2.2e+4) -	<b>5.19e-1 (1.0e+0) +</b>	6.24e-1 (1.4e-1)
MaF4	3	3.03e+0 (3.1e+0) -	2.36e+0 (5.2e+0) -	<b>2.88e-1 (2.0e-2) =</b>	3.06e-1 (1.7e-2) -	4.87e-1 (4.3e-1) -	2.94e-1 (2.4e-2)
	5	2.79e+3 (6.8e+2) -	1.23e+1 (2.8e+1) -	2.69e+0 (3.4e-1) -	2.71e+0 (5.2e-1) -	4.17e+0 (9.7e-1) -	<b>2.12e+0 (6.6e-2)</b>
	8	3.42e+4 (1.0e+4) -	9.74e+1 (1.6e+2) -	2.10e+1 (2.2e+0) =	2.29e+1 (2.2e+0) -	4.46e+1 (9.3e+0) -	<b>2.03e+1 (1.3e+0)</b>
	10	1.12e+5 (2.9e+4) -	2.62e+2 (3.6e+2) -	7.86e+1 (5.4e+0) +	<b>7.32e+1 (8.9e+0) +</b>	1.97e+2 (5.7e+1) -	8.97e+1 (9.7e+0)
MaF5	3	1.30e+0 (1.1e+0) -	5.96e-1 (3.5e-1) -	9.52e-1 (9.6e-1) -	4.92e-1 (5.7e-1) -	<b>2.23e-1 (3.5e-4) +</b>	2.72e-1 (2.3e-1)
	5	6.62e+0 (8.0e-1) -	1.27e+1 (4.7e-1) -	3.38e+0 (1.5e+0) -	2.21e+0 (6.8e-1) +	<b>2.11e+0 (8.3e-2) +</b>	2.21e+0 (7.5e-1)
	8	4.28e+1 (4.2e+0) -	8.24e+1 (5.7e+0) -	2.11e+1 (9.2e+0) =	<b>1.55e+1 (2.0e-1) =</b>	1.85e+1 (6.1e-1) =	2.31e+1 (1.5e+1)
	10	1.47e+2 (1.6e+1) -	2.92e+2 (1.5e+1) -	7.38e+1 (2.7e+1) =	<b>6.08e+1 (1.7e+0) =</b>	8.88e+1 (4.7e+0) =	8.91e+1 (4.6e+1)
MaF6	3	7.26e-3 (6.5e-4) -	1.09e-2 (1.9e-4) -	6.13e-2 (1.3e-2) -	1.04e-2 (1.8e-3) -	3.50e-2 (5.2e-3) -	<b>5.85e-3 (3.7e-4)</b>
	5	4.20e+0 (2.5e+0) -	2.53e-2 (2.7e-4) -	5.66e-2 (1.4e-2) -	2.53e-2 (1.1e-2) -	7.20e-2 (2.4e-2) -	<b>7.85e-3 (1.1e-3)</b>
	8	1.55e+1 (8.7e+0) -	4.32e-2 (3.0e-4) -	5.71e-2 (1.5e-2) -	2.63e-1 (2.1e-1) -	1.08e-1 (1.5e-2) -	<b>8.64e-3 (1.7e-3)</b>
	10	1.18e+1 (6.5e+0) -	5.34e-2 (9.4e-4) -	5.31e-2 (1.2e-2) -	5.03e-1 (2.1e-1) -	1.18e-1 (1.4e-2) -	<b>2.35e-2 (5.1e-2)</b>
MaF7	3	1.62e-1 (1.2e-1) -	2.51e-1 (7.5e-2) -	1.21e-1 (4.6e-2) -	6.61e-2 (4.4e-3) -	9.29e-2 (3.6e-3) -	<b>5.87e-2 (1.9e-3)</b>
	5	5.58e+0 (1.4e+0) -	1.60e+0 (3.1e-1) -	3.45e-1 (6.9e-2) -	3.09e-1 (7.7e-3) -	4.16e-1 (6.2e-3) -	<b>2.86e-1 (6.0e-3)</b>
	8	2.53e+1 (3.6e+0) -	4.79e+0 (4.9e-1) -	7.85e-1 (7.0e-2) -	6.80e-1 (8.3e-3) -	1.02e+0 (4.5e-2) -	<b>6.23e-1 (1.1e-2)</b>
	10	3.04e+1 (4.5e+0) -	6.90e+0 (4.6e-1) -	1.05e+0 (3.5e-2) -	9.61e-1 (2.8e-2) -	1.47e+0 (1.3e-1) -	<b>7.18e-1 (6.6e-3)</b>

MaF8	3	1.27e-1 (3.2e-2) -	9.03e-2 (1.3e-3) +	8.20e-2 (4.4e-3) =	8.39e-2 (7.1e-3) =	1.21e-1 (7.8e-3) -	1.02e-1 (6.6e-2)
	5	8.76e+1 (6.2e+1) -	1.01e-1 (1.2e-3) +	1.22e-1 (4.1e-3) +	1.49e-1 (8.0e-3) =	3.70e-1 (3.7e-2) -	1.48e-1 (3.9e-2)
	8	4.07e+2 (2.3e+2) -	1.33e-1 (1.4e-3) +	1.48e-1 (4.9e-3) +	3.01e-1 (2.8e-2) +	6.95e-1 (8.1e-2) -	3.69e-1 (4.6e-2)
	10	3.60e+2 (2.1e+2) -	1.24e-1 (9.8e-4) +	1.36e-1 (3.5e-3) +	2.93e-1 (4.5e-2) +	7.86e-1 (7.1e-2) -	5.25e-1 (5.3e-2)
MaF9	3	1.08e-1 (2.8e-2) -	7.57e-2 (6.7e-4) -	2.58e-1 (1.2e-1) -	6.22e-2 (5.0e-3) =	1.10e-1 (1.0e-1) =	6.94e-2 (1.8e-2)
	5	5.01e+1 (3.3e+1) -	1.18e-1 (1.4e-3) -	2.17e-1 (4.5e-2) -	5.56e-1 (2.1e-1) -	3.18e-1 (3.4e-2) -	1.17e-1 (3.8e-2)
	8	2.21e+2 (1.5e+2) -	1.54e-1 (4.2e-3) =	7.80e-1 (1.9e-1) -	4.03e-1 (2.9e-1) -	5.79e-1 (8.3e-2) -	1.83e-1 (6.7e-2)
	10	1.26e+2 (8.1e+1) -	1.56e-1 (2.9e-3) +	9.99e-1 (6.5e-2) -	3.14e-1 (4.9e-2) +	9.46e-1 (3.0e-1) -	4.71e-1 (9.6e-1)
MaF10	3	4.53e-1 (9.2e-2) -	1.09e+0 (1.5e-1) -	4.64e-1 (1.4e-1) -	1.98e-1 (2.4e-2) +	2.96e-1 (4.0e-2) +	3.43e-1 (5.0e-2)
	5	2.30e+0 (6.9e-2) -	1.79e+0 (1.6e-1) -	9.22e-1 (1.2e-1) =	6.38e-1 (7.7e-2) +	5.09e-1 (5.8e-2) +	9.36e-1 (7.0e-2)
	8	3.04e+0 (8.0e-2) -	2.67e+0 (1.4e-1) -	1.83e+0 (1.7e-1) +	9.54e-1 (6.7e-2) +	8.94e-1 (3.2e-2) +	1.90e+0 (4.9e-2)
	10	3.38e+0 (6.0e-2) -	2.68e+0 (1.1e-1) -	2.02e+0 (1.6e-1) +	1.34e+0 (1.0e-1) +	1.13e+0 (4.5e-2) +	2.13e+0 (9.2e-2)
MaF11	3	1.75e-1 (9.0e-3) =	3.15e-1 (1.5e-2) -	2.29e-1 (2.2e-2) -	1.49e-1 (4.4e-3) +	1.70e-1 (7.2e-3) =	1.87e-1 (2.7e-2)
	5	1.06e+0 (9.1e-2) -	1.16e+0 (1.3e-1) -	7.20e-1 (3.6e-2) +	4.31e-1 (1.7e-2) +	4.06e-1 (7.1e-3) +	8.44e-1 (7.9e-2)
	8	3.00e+0 (1.4e+0) =	2.63e+0 (7.3e-1) -	1.31e+0 (1.1e-1) +	9.64e-1 (2.1e-1) +	9.15e-1 (2.8e-2) +	2.19e+0 (1.0e-1)
	10	3.48e+0 (1.0e+0) -	4.78e+0 (2.0e+0) -	1.70e+0 (2.5e-1) +	1.27e+0 (1.6e-1) +	1.05e+0 (2.5e-2) +	2.28e+0 (7.5e-2)
MaF12	3	2.29e-1 (3.6e-2) =	3.10e-1 (4.2e-2) -	2.99e-1 (5.6e-2) -	2.23e-1 (2.4e-2) =	1.98e-1 (3.3e-3) +	2.19e-1 (6.0e-3)
	5	1.84e+0 (1.2e-1) -	2.57e+0 (1.8e-1) -	1.19e+0 (4.8e-2) -	1.03e+0 (8.0e-3) -	1.03e+0 (2.3e-3) -	1.01e+0 (6.8e-3)
	8	5.05e+0 (6.0e-1) -	5.68e+0 (2.9e-1) -	3.45e+0 (2.1e-1) -	2.83e+0 (9.9e-3) -	2.89e+0 (2.1e-2) -	2.80e+0 (2.1e-2)
	10	7.27e+0 (7.3e-1) -	8.34e+0 (5.8e-1) -	5.30e+0 (4.3e-1) -	3.88e+0 (1.9e-2) =	4.23e+0 (4.7e-2) -	3.87e+0 (2.2e-2)
+/-/=		2/43/3	7/40/1	11/28/9	18/23/7	14/29/5	