

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

HEBER VALDO NOGUEIRA

Algoritmo Genético Compacto com Dominância para Seleção de Variáveis

Goiânia
2017

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR AS TESES E DISSERTAÇÕES ELETRÔNICAS NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei nº 9610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou *download*, a título de divulgação da produção científica brasileira, a partir desta data.

1 **1. Identificação do material bibliográfico:** ☒ **Dissertação** ☐ **Tese**

1 **2. Identificação da Tese ou Dissertação**

2


Nome completo do autor: Heber Valdo Nogueira

Título do trabalho: Algoritmo Genético Compacto com Dominância para Seleção de Variáveis

3. Informações de acesso ao documento:

Concorda com a liberação total do documento ☒ SIM ☐ NÃO¹

Havendo concordância com a disponibilização eletrônica, torna-se imprescindível o envio do(s) arquivo(s) em formato digital PDF da tese ou dissertação.



Data: 19 / 05 / 2017

¹ Neste caso o documento será embargado por até um ano a partir da data de defesa. A extensão deste prazo suscita justificativa junto à coordenação do curso. Os dados do documento não serão disponibilizados durante o período de embargo.

HEBER VALDO NOGUEIRA

Algoritmo Genético Compacto com Dominância para Seleção de Variáveis

Dissertação apresentada ao Programa de Pós-Graduação do Instituto de Informática da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Ciência da Computação.

Orientador: Prof. Dr. Anderson da Silva Soares

Co-Orientadora: Profa. Dra. Telma Woerle de Lima Soares

Goiânia
2017

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

' Algoritmo Genético Compacto com Dominância para Seleção de
Variáveis [manuscrito] / , Heber Valdo Nogueira. - 2017.
f.: il.

Orientador: Prof. Dr. Anderson da Silva Soares; co-orientadora
Dra. Telma Woerle Lima Soares.

Dissertação (Mestrado) - Universidade Federal de Goiás, Instituto
de Informática (INF), Programa de Pós-Graduação em Ciência da
Computação, Goiânia, 2017.

Bibliografia.

Inclui abreviaturas, símbolos, gráfico, tabelas, algoritmos, lista de
figuras, lista de tabelas.

1. Seleção de variáveis. 2. Algoritmo Genético Compacto. 3.
Otimização multiobjetivo. I. da Silva Soares, Anderson , orient. II. Título.

CDU 004



ATA Nº 08/2017

**ATA DA SESSÃO DE JULGAMENTO DA DISSERTAÇÃO
DE MESTRADO DE HEBER VALDO NOGUEIRA**

Aos vinte dias do mês de abril de dois mil e dezessete, às catorze horas e trinta minutos, na sala 150 do Instituto de Informática da Universidade Federal de Goiás, Campus Samambaia, reuniu-se a banca examinadora designada na forma regimental pela Coordenação do Curso para julgar a dissertação de mestrado intitulada “**Algoritmo Genético Compacto com Dominância para Seleção de Variáveis**”, apresentada pelo aluno Heber Valdo Nogueira como parte dos requisitos necessários à obtenção do grau de Mestre em Ciência da Computação, área de concentração Ciência da Computação. A banca examinadora foi presidida pelo orientador do trabalho de dissertação, Professor Doutor Anderson da Silva Soares (INF/UFG), tendo como membros os Professores Doutores Telma Woerle de Lima Soares (INF/UFG – coorientadora), Clarimar José Coelho (PUC/GO) e Jailson Cardoso Dias (Souza Cruz S.A.). Aberta a sessão, o candidato expôs seu trabalho. Em seguida, o aluno foi arguido pelos membros da banca e:

☒ tendo demonstrado suficiência de conhecimento e capacidade de sistematização do tema de sua dissertação, a banca concluiu pela **aprovação** do candidato, sem restrições.

() não tendo demonstrado suficiência de conhecimento e capacidade de sistematização do tema de sua dissertação, a banca concluiu pela **reprovação** do candidato.

Os trabalhos foram encerrados às 16:53 horas. Nos termos do Regulamento Geral dos Cursos de Pós-Graduação desta Universidade, lavrou-se a presente ata que, lida e julgada conforme, segue assinada pelos membros da banca examinadora.

Prof. Dr. Anderson da Silva Soares _____
Profa. Dra. Telma Woerle de Lima Soares _____
Prof. Dr. Clarimar José Coelho _____
Prof. Dr. Jailson Cardoso Dias _____

Dedico este trabalho especialmente aos meus pais e a minha família.

Agradecimentos

Primeiramente, agradeço a Deus pela vida e por sempre me guiar na direção correta, provando que Teus planos são sempre melhores que os meus.

Agradeço imensamente aos meus pais, José Valdo Nogueira e Jucê Batista Nogueira, por perdoarem minhas falhas, pela cumplicidade e por sempre acreditarem na minha capacidade.

A minha família, pelo apoio irrestrito, pela compreensão e por ser minha fonte inesgotável de amor e carinho.

A minha namorada, Lorena Pereira de Sousa Rosa, por sua compreensão e incentivo, além da parceria nas madrugadas de estudos.

Ao meu amigo Prof. Msc Arlindo Rodrigues Galvão Filho, pelo incentivo e por ser meu parceiro de estudos e pesquisas há mais de 6 anos.

Ao meu orientador, Prof. Dr. Anderson da Silva Soares, por sua paciência e amizade. Por ser um dos maiores pesquisadores residentes em Goiás e, ainda assim, manter a simplicidade e a capacidade de empolgar seus alunos.

A minha co-orientadora, Profa. Dra. Telma Woerle de Lima Soares, por suas ideias brilhantes e por sempre me receber de braços abertos em sua residência.

Ao Prof. Dr. Clarimar José Coelho, pela sua amizade, conselhos e paixão pela ciência.

Aos amigos que fiz em São José dos Campos, por terem sido uma família durante a época que estive por lá e por terem ajudado na minha formação como pesquisador.

Aos meus amigos que sempre me apoiaram na busca pelo aperfeiçoamento intelectual.

Aos meus colegas da Assembleia Legislativa de Goiás, pela parceria, incentivo e sonhos compartilhados.

Por fim, agradeço a todos que, de alguma forma, participaram da minha vida acadêmica durante os últimos cinco anos.

"O insucesso é apenas uma oportunidade para recomeçar de novo com mais inteligência"

Henry Ford,
fundador da Ford Motor Company.

Resumo

Nogueira, Heber V.. **Algoritmo Genético Compacto com Dominância para Seleção de Variáveis**. Goiânia, 2017. 61p. Dissertação de Mestrado. Instituto de Informática, Universidade Federal de Goiás.

O problema de seleção de variáveis consiste em selecionar um subconjunto de atributos que seja capaz de reduzir os recursos computacionais de processamento e armazenamento, diminuir os efeitos da maldição da dimensionalidade e melhorar a performance de modelos de predição. Dentre as estratégias utilizadas nesse tipo de problema, destacam-se os algoritmos evolutivos, como o Algoritmo Genético. Apesar do relativo sucesso do Algoritmo Genético na solução de variados tipos de problemas, diferentes propostas de melhoria têm sido apresentadas no sentido de aprimorar seu desempenho. Tais melhorias focam, sobretudo, na representação da população, nos mecanismos de busca e nos métodos de avaliação. Em uma dessas propostas, surgiu o Algoritmo Genético Compacto (AGC), que propõe novas formas de representar a população e de conduzir a busca por melhores soluções. A aplicação desse tipo de estratégia para solucionar o problema de seleção de variáveis pode implicar no *overfitting*. Diversas pesquisas na área têm indicado que a abordagem multiobjetivo pode ser capaz de mitigar esse tipo de problema. Nesse contexto, este trabalho propõe a implementação de três versões do Algoritmo Genético Compacto capazes de minimizar mais de um objetivo simultaneamente. O primeiro faz uso do conceito de dominância de Pareto e, por isso, é chamado de Algoritmo Genético Compacto com Dominância (AGC-D). Os outros dois utilizam estratégias de ponderação das funções objetivo para estabelecer prioridades durante a busca. Como estudo de caso, para avaliar o desempenho do algoritmo proposto, os algoritmos são combinados com a Regressão Linear Múltipla com o objetivo de selecionar variáveis para melhor prever a concentração de proteína em amostras de trigo. Os algoritmos propostos são comparados entre si, ao AGC e ao AGC com operador de mutação. Os resultados obtidos indicam que o AGC-D é capaz de selecionar um pequeno conjunto de variáveis, reduzindo o erro de predição do modelo de calibração e minimizando a ocorrência de *overfitting*.

Palavras-chave

seleção de variáveis, algoritmo genético compacto, otimização multiobjetivo.

Abstract

Nogueira, Heber V.. **Compact Genetic Algorithm with Dominance for Feature Selection**. Goiânia, 2017. 61p. MSc. Dissertation. Instituto de Informática, Universidade Federal de Goiás.

The features selection problem consists in to select a subset of attributes that is able to reduce computational processing and storage resources, decrease curse of dimensionality effects and improve the performance of predictive models. Among the strategies used to solve this type of problem, we highlight evolutionary algorithms, such as the Genetic Algorithm. Despite the relative success of the Genetic Algorithm in solving various types of problems, different improvements have been proposed in order to improve their performance. Such improvements focus mainly on population representation, search mechanisms, and evaluation methods. In one of these proposals, the Compact Genetic Algorithm (CGA) arose, which proposes new ways of representing the population and guide the search for better solutions. Applying this type of strategy to solve the problem of variable selection often involves overfitting. In this context, this work proposes the implementation of three versions of the Compact Genetic Algorithm capable of minimizing more than one objective simultaneously. The first makes use of the concept of Pareto dominance and, therefore, is called the Compact Genetic Algorithm with Dominance (CGA-D). The other two use weighting strategies to establish priorities during the search. As a case study, to evaluate the performance of the proposed algorithms, the algorithms are combined with Multiple Linear Regression (MLR) to select variables to better predict protein concentration in wheat samples. The proposed algorithms are compared between them and to CGA and Mutation-based Compact Genetic Algorithm. The results indicate that the CGA-D is able to select a small set of variables, reducing the prediction error of the calibration model, reducing the possibility of overfitting.

Keywords

feature selection, compact genetic algorithm, multiobjective optimization

Sumário

Lista de Figuras	10
Lista de Tabelas	11
Lista de Algoritmos	12
Lista de Smbolos	13
1 Introdução	14
2 Fundamentação Teórica	18
2.1 Introdução	18
2.2 Algoritmo Genético	18
2.2.1 População	20
2.2.2 Seleção	22
2.2.3 Cruzamento	22
2.2.4 Mutação	23
2.3 Hipótese dos <i>Building Blocks</i>	23
2.3.1 Teorema dos Esquemas	23
2.3.2 <i>Building Blocks</i>	25
2.4 Algoritmo Genético Compacto	26
2.5 Otimização Multiobjetivo	28
2.5.1 Dominância de Pareto	29
2.5.2 Método da Soma Ponderada	31
3 Implementações Propostas	35
3.1 Algoritmo Genético Compacto com Dominância	35
3.2 Algoritmo Genético Compacto com Ponderação	38
3.3 Algoritmo Genético Compacto com Soma Ponderada	39
4 Material e Métodos	41
4.1 Estudo de caso	41
4.2 Conjunto de Dados	42
4.3 Forma de Análise dos Resultados	43
4.4 Plataforma Computacional	44

5	Resultados	45
5.1	Determinação do tamanho da população	45
5.2	Resultados para o AGC-D	47
5.3	Resultados para as versões ponderadas do AGC	49
5.4	Avaliação da Robustez dos Algoritmos Propostos	52
6	Conclusões	55
6.1	Trabalhos Futuros	55
	Referências Bibliográficas	57

Lista de Figuras

2.1	Representação de um indivíduo.	20
2.2	Soluções para o problema de minimização do custo do <i>hardware</i> e maximização de seu desempenho.	28
2.3	Ilustração do conceito de soluções Pareto-ótimas.	30
2.4	Representação do espaço de busca de um problema de minimização com dois objetivos.	33
2.5	Representação do espaço de busca não convexo.	34
3.1	Exemplo de funcionamento do AGC-D: (a) exemplifica o caso quem que uma solução foi melhor nos dois objetivos avaliados; (b) exemplifica o caso quem que uma solução foi melhor em no primeiro objetivo e a outra solução foi melhor no segundo objetivo.	37
	(a)	37
	(b)	37
4.1	Espectro derivativo das 775 amostras e trigo.	43
5.1	Quantidade de variáveis selecionadas para diferentes tamanhos de população simulada.	45
5.2	Erro de predição para diferentes tamanho de população simulada.	46
5.3	Nível de parcimônia entre a quantidade de variáveis selecionadas e o erro de predição para diferentes tamanhos de população.	47
5.4	Variáveis selecionadas pelo AGC.	48
5.5	Variáveis selecionadas pelo AGC-D.	49
5.6	Comparação entre os algoritmos AGC-P e AGC-SP.	50
5.7	Comparação entre os algoritmos AGC-D e AGC-SP.	51
5.8	Avaliação do AGC na presença de dados ruidosos.	52
5.9	Avaliação do AGC-SP na presença de dados ruidosos.	53
5.10	Avaliação do AGC-D na presença de dados ruidosos	53

Lista de Tabelas

5.1	Desempenho dos algoritmos em relação à quantidade de variáveis selecionadas e ao erro de predição.	47
5.2	Desempenho dos algoritmos AGC-P e AGC-SP em relação à quantidade de variáveis selecionadas e ao erro de predição.	50
5.3	Desempenho dos algoritmos AGC-D e AGC-SP em relação à quantidade de variáveis selecionadas e ao erro de predição.	51

Lista de Algoritmos

2.1	Pseudocódigo do AG clássico.	19
2.2	Algoritmo Genético Compacto, adaptado de Harik <i>et al.</i> [26].	27
3.1	Algoritmo Genético Compacto com Dominância - AGC-D	36
3.2	Algoritmo Genético Compacto com Ponderação - AGC-P	38
3.3	Algoritmo Genético Compacto com Soma Ponderada - AGC-SP	39

Lista de Símbolos

\mathbf{x}	indivíduo, solução ou vetor de variáveis independentes	14
y	variável dependente	14
l	comprimento do indivíduo ou número de variáveis	14
\mathbf{X}	Espaço de busca	18
\mathbf{x}^*	solução ótima	18
f	função objetivo	18
\mathbf{P}	Conjunto de indivíduos	19
Φ_g	Genótipo	24
Φ_f	Fenótipo	21
f_g	Função que mapeia o genótipo em fenótipo	21
f_f	Função que mapeia o fenótipo em um valor no plano Real	21
t	Quantidade de torneios realizados	22
j	Quantidade de indivíduos disputando um torneio	22
σ	Alfabeto	24
\mathbf{h}	Esquema	24
$O(h)$	Ordem do esquema h	24
$\delta(h)$	Comprimento definitivo do esquema h	24
\mathbf{p}	Vetor de probabilidades	26
n	Tamanho da população simulada	27
s	Quantidade de objetivos a serem otimizados	29
w	Coefficiente de ponderação	44
m	Número de amostras de grãos de trigo	41
\mathbf{y}	vetor de variáveis dependentes	41
$\hat{\mathbf{y}}$	vetor de variáveis dependentes estimado	42
β	Vetor dos coeficientes de regressão	41
ε	Parcela de erro aleatório	41
λ	Comprimento de onda	43
$RMSEP$	Erro de predição do modelo de calibração	44
QV	Quantidade de variáveis selecionadas	44
n_e	Quantidade de simulações realizadas	46

Introdução

Uma variável é uma propriedade individual que pode ser mensurada a partir de um problema considerado. Algoritmos de aprendizado de máquina, tais como os de regressão, classificação e agrupamento, fazem uso de variáveis informativas para solucionar problemas de reconhecimento de padrões. Diversos problemas do mundo real possuem um número elevado de variáveis relacionadas ao objeto de estudo. Considere, por exemplo, y como a variável dependente e $\mathbf{x} = x_1, x_2, \dots, x_l$ um conjunto de variáveis independentes. Idealmente, o melhor subconjunto de variáveis seria encontrado após avaliar cada um dos 2^l subconjuntos possíveis. Contudo, para grandes valores de l , tal tarefa pode ser computacionalmente proibitiva. Logo, estratégias de seleção de subconjunto de variáveis são necessárias para reduzir os recursos computacionais de processamento e armazenamento de dados, diminuir os efeitos da maldição da dimensionalidade e melhorar a performance de modelos de predição [1, 6].

Selecionar o subconjunto de dados mais apropriados para a solução de um problema nem sempre é uma tarefa trivial, especialmente pela extensão do espaço de busca. Diante da constante evolução dos métodos de aquisição de dados, a busca exaustiva tem se tornado impraticável [6]. Adicionalmente, problemas como a multicolinearidade podem dificultar ainda mais a determinação do subconjunto de dados que melhor modela a relação entre as variáveis dependentes e independentes [20]. Estes e outros problemas têm influenciado no desenvolvimento de novas estratégias para seleção de variáveis [1, 9].

As estratégias de seleção de variáveis são geralmente divididas em duas classes: *filters* e *wrappers* [30]. As estratégias classificadas como *filters* utilizam métodos estatísticos para ranquear as variáveis de acordo com seu potencial de informação. Além de serem mais rápidos computacionalmente, os *filters* são capazes de eliminar variáveis redundantes evitando problemas de multicolinearidade. O algoritmo Relief, desenvolvido por Kira e Rendell [29], por exemplo, faz uso da distância euclidiana para mensurar a relevância das variáveis de um conjunto e, posteriormente, selecionar aquelas que possuem nível de relevância acima de um determinado limiar. Contudo, ao avaliar individualmente cada variável, os *filters* acabam desprezando a existência de interação entre elas, fenômeno conhecido como *building block*. Tal característica pode ocasionar a eliminação de

variáveis que individualmente são pouco representativas, mas que em um subconjunto podem melhorar a acurácia do classificador [1, 6]. As estratégias do tipo *wrapper*, por sua vez, possuem dois pontos chaves: a técnica de seleção de variáveis e o critério de avaliação. Enquanto o primeiro explora o espaço de busca para encontrar o melhor subconjunto de dados, o segundo utiliza um classificador para mensurar a qualidade do subconjunto selecionado. Em Goodman *et al.* [49] e Yang [52] uma implementação de um Algoritmo Genético (AG) é utilizada como estratégia de busca e seleção para os classificadores KNN (*K-Nearest Neighbor*) e Redes Neurais Artificiais, respectivamente. Ambos os classificadores avaliam cada subconjunto de variáveis selecionadas pelo AG de modo a guiar o processo de busca e seleção. Em razão desta combinação, as abordagens tipo *wrappers* geralmente apresentam melhor desempenho nos algoritmos de classificação. As principais desvantagens desse tipo de estratégia são o alto custo computacional para realizar a avaliação por meio do teste de classificação de cada novo subconjunto de dados e a propensão dos classificadores a causar o *overfitting*¹ [23].

Dentre as técnicas de seleção em estratégias do tipo *wrapper*, é possível destacar os algoritmos evolutivos ou genéticos. Siedlecki e Sklansky [45] apresentaram um dos trabalhos pioneiros deste tipo de algoritmo para o problema de seleção de variáveis. Desde então, diferentes propostas têm sido apresentadas no sentido de aprimorar o seu desempenho e adaptá-las às especificidades de diferentes aplicações. Tais melhorias focam, sobretudo, na representação da população, nos mecanismos de busca e nos métodos de avaliação, que segundo Rothlauf [42] são alguns dos pontos principais no projeto de algoritmos evolutivos. Estes elementos possuem as seguintes definições:

- Representação: descreve como uma solução ao problema considerado pode ser codificada na implementação do algoritmo.
- Mecanismo de busca: descreve como será implementado as operações, tais como, recombinação e seleção, que visam modificar as soluções que existem de modo a explorar o espaço de busca;
- Método de avaliação: tem por objetivo quantificar a qualidade de uma determinada solução no contexto do problema considerado.

Uma quantidade significativa de trabalhos, tais como o de Yang e Hanavar [52], utilizam como representação cromossomos do tipo binário em conjunto com mecanismos de busca típicos desta representação, tais como, recombinação e mutação. Apesar de seu amplo uso e relativo sucesso na solução de uma grande variedade de problemas, alguns

¹O *overfitting* ocorre quando são utilizadas mais variáveis que o necessário para descrever a relação entre as variáveis dependentes e independentes. Como consequência, o classificador perde capacidade de generalização.

trabalhos indicam que o desempenho desse tipo de algoritmo carece de aprimoramentos [40, 18]. Um ponto a se destacar em relação ao Algoritmo Evolutivo com codificação binária é, por exemplo, a escolha da quantidade de pontos de cruzamento. Tsuji *et al.* [37] cita que os operadores de cruzamento podem romper os *building blocks* [22, 36]. Como exemplo é possível citar o trabalho de Paula *et al.* [12] que mostrou empiricamente que para o caso de um problema de seleção de variáveis envolvendo dados de espectrofotômetros, os algoritmos evolutivos em sua codificação binária e operadores padrões apresentam rompimento de *building blocks*.

Com o objetivo de oferecer uma abordagem alternativa para o tratamento dos *building blocks*, uma nova classe de Algoritmos Evolutivos tem sido investigada: os Algoritmos de Estimação de Distribuição (AED). Os AEDs diferem dos AEs tradicionais nas maneiras de representar a população e no funcionamento dos mecanismos de busca. Enquanto os AEs tradicionais utilizam operadores de cruzamento e mutação, os AEDs geram novas soluções a partir de modelos probabilísticos. Esta estratégia permite que cada variável seja avaliada de maneira independente. Dessa maneira, enquanto os AEs tratam os *building blocks* de maneira implícita, os AEDs explicitam a interação entre tais blocos utilizando a distribuição de probabilidade associada a cada indivíduo da população [32].

Um exemplo de algoritmo pertencente à classe dos AEDs é o Algoritmo Genético Compacto (AGC). Introduzido por Harik *et al.* [26], o AGC utiliza um vetor de probabilidades para representar a população e gerar novas soluções. O mecanismo de busca do algoritmo atualiza bit a bit o vetor de probabilidades a cada geração, aumentando as chances de reprodução dos *building blocks*. Em Soares *et al.* [46] uma versão do AGC com operador de mutação foi apresentada com o objetivo de selecionar variáveis em um problema de calibração multivariada. Os resultados obtidos indicam que, para o problema em questão, a abordagem utilizada pelo AGC para manipular os *building blocks* apresenta melhor desempenho quando comparado ao AG clássico. Contudo, apesar dos bons resultados de predição obtidos, o grande número de variáveis selecionadas pelo algoritmo pode provocar o *overfitting*.

Assim como em Soares *et al.* [46], diversas outras estratégias de seleção de variáveis tipo *wrapper* utilizam apenas a acurácia do classificador, ou regressor, como critério de avaliação. Conhecida como monoobjetivo, esta abordagem permite que qualquer variável que melhore a acurácia do classificador seja adicionada ao modelo de predição. Nesse contexto, um número relativamente grande de variáveis pode ser adicionada ao modelo mesmo que o ganho em acurácia seja mínimo. Uma alternativa frequentemente utilizada para contornar este problema é utilizar a abordagem multiobjetivo como critério de avaliação. Neste caso, por exemplo, é possível utilizar a capacidade preditiva do modelo obtido e a quantidade de variáveis utilizadas. Em Ribeiro *et al.* [10], a versão multiobjetivo do AG é aplicada em um problema de seleção de variáveis em calibração multivariada com

o objetivo de detectar adulteração em amostras de biodiesel. Em Lucena *et al.* [11] e Campos Jorge [28] algoritmos evolutivos multiobjetivo são utilizado para determinar um subconjunto de variáveis descritivas e aperfeiçoar modelos de predição. O primeiro usa os algoritmos genéticos NSGA-II e o SPEA-II enquanto o segundo utiliza um Algoritmo Evolutivo Multiobjetivo com Tabelas. Em ambos os casos os resultados apresentados indicam um desempenho superior da abordagem multiobjetivo em relação a algoritmos com formulação monoobjetivo.

Considerando o exposto, o objetivo deste trabalho é apresentar uma versão do Algoritmo Genético Compacto capaz de minimizar, simultaneamente, mais de um objetivo. Tal algoritmo é denominado Algoritmo Genético Compacto com Dominância (AGC-D). Como estudo de caso, o AGC-D é combinado com a Regressão Linear Multipla (RLM) com o objetivo de selecionar variáveis para melhor prever a concentração de proteína em amostras de trigo, minimizando o erro de predição do modelo e a quantidade de variáveis selecionadas. Em um primeiro momento, o algoritmo proposto é comparado ao AGC monoobjetivo e à versão do AGC com operador de mutação (*Mutation-based Compact Genetic Algorithm - mCGA* apresentado em Soares *et al.* [46]. Posteriormente, duas versões ponderadas do AGC são apresentadas com a finalidade de possibilitar o estabelecimento de prioridades entre os objetivos avaliados. Por fim, é avaliada a robustez dos algoritmos propostos.

Na sequência deste trabalho, o Capítulo 2 apresenta conceitos básicos do Algoritmo Genético clássico e do Algoritmo Genético Compacto, além de uma discussão sobre *building blocks* e fundamentos de otimização multiobjetivo. No Capítulo 3 são apresentadas as implementações propostas. No Capítulo 4 são apresentados o estudo de caso, o conjunto de dados e os métodos computacionais utilizados para avaliar os algoritmos propostos. Os resultados são apresentados e discutidos no Capítulo 5. Por fim, o Capítulo 6 conclui o trabalho e apresenta os trabalhos futuros.

Fundamentação Teórica

Neste capítulo apresenta-se as formulações básicas dos Algoritmos Genéticos clássico e compacto e a formulação da hipótese dos *building blocks*. Por fim, realiza-se uma descrição do problema de otimização multiobjetivo sob a ótica do conceito de dominância e do método de soma ponderada.

2.1 Introdução

O processo de otimização consiste em encontrar soluções que sejam ótimas, ou sub-ótimas, com respeito a determinados objetivos. Formalmente, um problema de otimização pode ser descrito da seguinte maneira [50]:

- Encontrar uma solução $\mathbf{x}_0 \in \mathbf{X}$ tal que f seja ótima em \mathbf{X} , sendo $f : \mathbf{X} \rightarrow \mathbb{R}$ uma função real valorada qualquer e \mathbf{X} é o espaço de busca.

Considerando problemas de otimização, o espaço de busca corresponde ao conjunto de possíveis soluções de um dado problema, onde f é a função que avalia a aptidão (ou *fitness*) de cada solução. A solução ótima de um problema de otimização é chamada de ótima global sendo definida como a solução $\mathbf{x}^* \in \mathbf{X}$ onde $f(\mathbf{x}^*) \leq f(\mathbf{x})$ para todo $\mathbf{x} \in \mathbf{X}$ (considerando um problema de minimização) [42].

Na prática, algumas vezes pode ser difícil determinar a solução ótima de um problema de otimização. Dependendo do problema, pode ser suficiente encontrar uma solução próxima da solução ótima, neste caso, chamada de subótima. Nesse contexto, diversos métodos de busca podem ser aplicados para encontrar tais soluções, com destaque para os algoritmos evolutivos, como o Algoritmo Genético (AG).

2.2 Algoritmo Genético

O Algoritmo Genético é um paradigma computacional utilizado em problemas de busca e otimização. A principal vantagem do AG é o fato de ser uma técnica de

propósito geral, o que o permite solucionar problemas em diversas áreas do conhecimento. Apesar de não garantir a determinação de uma solução ótima, o AG é capaz de encontrar soluções subótimas em um tempo computacionalmente aceitável [4].

Inspirado na teoria da seleção natural de Darwin, o AG utiliza princípios de sobrevivência e reprodução dos indivíduos mais aptos para produzir melhores soluções a cada geração [34]. O processo de busca dos AGs é estocástico e iterativo. Assim como em diversas outras estratégias evolutivas, os principais componentes do AG são [19]:

- População de indivíduos: uma ou mais populações que concorrem por recursos limitados;
- Aptidão: reflete a habilidade de um indivíduo de sobreviver e reproduzir-se;
- Dinâmica da mudança nas populações: que ocorre devido ao nascimento e morte dos indivíduos;
- Conceitos de variabilidade e hereditariedade: indivíduos de novas gerações possuem muitas das características de seus antecedentes, embora não sejam idênticos.

Para um indivíduo em particular, a combinação de genes é chamada de genótipo e as características do indivíduo, baseadas no genótipo, constituem o fenótipo. Portanto, quando indivíduos é avaliado, tal avaliação é feita em nível de fenótipo. Todavia, quando é realizada a reprodução, tal reprodução é realizada em nível de genótipo.

Em sua formulação clássica [27], o Algoritmo Genético trabalha com uma população de indivíduos \mathbf{P} , onde cada indivíduo \mathbf{x} equivale a uma possível solução no espaço de busca. Cada indivíduo da população é avaliado por uma função objetivo f que mede seu nível de aptidão. O pseudocódigo do AG é apresentado no Algoritmo 2.1.

Algoritmo 2.1: Pseudocódigo do AG clássico.

1. $ger \leftarrow 0$; //Inicializa o contador de gerações
 2. $\mathbf{P}(ger) \leftarrow \text{Inicializa_populacao}()$; //Gera aleatoriamente uma população inicial
 3. **Enquanto** $g < \text{criterio_parada}$
 4. $\text{Avalia}(\mathbf{P}(ger))$; //Avalia cada indivíduo da população inicial de acordo com a função objetivo f
 5. $(\mathbf{x}_1, \mathbf{x}_2) \leftarrow \text{Selecao_individuos}()$; //Seleção de dois indivíduos para reprodução de descendentes
 6. $\mathbf{x}_c \leftarrow \text{cruzamento}(\mathbf{x}_1, \mathbf{x}_2)$; //Cruzamento dos pais para gerar os filhos
 7. $\mathbf{x}_m \leftarrow \text{mutacao}(\mathbf{x}_c)$; //operação de mutação dos filhos gerados
 8. $ger \leftarrow ger + 1$;
 9. $\mathbf{P}(ger) \leftarrow \text{atualiza}(\mathbf{x}_m)$; //atualiza a População
 10. **Fim Enquanto**
-

O pseudocódigo representado no Algoritmo 2.1 começa tendo o contador de gerações inicializado e, em seguida, a população inicial \mathbf{P} é gerada. A terceira linha

do Algoritmo estabelece o critério de parada, utilizado para interromper a execução quando o limite máximo de gerações é atingido ou quando a população converge. Pode-se dizer que a população convergiu, quando pelo menos 95% dos alelos de cada indivíduo da população apresentam o mesmo valor [4]. Na linha 4, cada indivíduo da população é avaliado de acordo com a função objetivo f . Indivíduos com maiores níveis de aptidão possuem maiores probabilidades de serem selecionados na linha 5 para o processo de reprodução presentes nas linhas 6 e 7. Durante tal processo, operadores de cruzamento e mutação são utilizados para gerar novos indivíduos que compartilham algumas características de seus ascendentes. Na linha 9 a população P é atualizada utilizando os novos indivíduos gerados. Todo o processo se repete até que um critério de parada seja alcançado [51]. Cada etapa do AG é melhor detalhada a seguir.

2.2.1 População

Cada indivíduo da população representa uma possível solução do problema e equivale a um cromossomo na biologia [16]. O cromossomo é descrito por uma sequência de alelos que armazena as características de um determinado indivíduo. Cada característica de um indivíduo é representada por um gene, que pode ser composto por um ou mais alelos. O alelo, por sua vez, é a menor unidade de informação em um cromossomo e é representado por um símbolo. Em uma representação binária, por exemplo, o alelo pode assumir apenas os valores 0 ou 1. A Figura 2.1 ilustra a diferença entre cromossomo, alelo e gene [43].

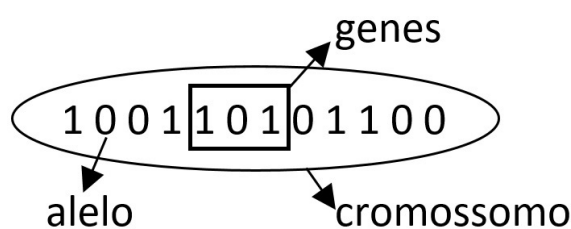


Figura 2.1: Representação de um indivíduo.

Uma etapa importante do projeto da população de um AG é a definição da representação das soluções, ou seja, como as soluções serão codificadas nos indivíduos. A utilização de um tipo de representação consiste na definição do genótipo e do mapeamento do genótipo em fenótipo. Para um indivíduo em particular, toda a informação armazenada nos cromossomos é representada pelo genótipo enquanto as características do indivíduo, baseadas no genótipo, são representadas pelo fenótipo [44].

No contexto do AG, o genótipo é a representação de uma solução a partir de uma determinada codificação. A definição do tipo de codificação utilizada na representação das soluções é crucial para a performance do processo de busca. Adicionalmente, dependendo do tipo de codificação escolhido, são necessários operadores de busca específicos [16, 19].

A codificação binária é a representação utilizada em AGs. Tal codificação foi originalmente aplicada por imitar a representação genética dos organismos vivos [53]. Formalmente, o genótipo binário é definido como um arranjo de 0's e 1's:

$$\phi_g = \{0, 1\}^l, \quad (2-1)$$

onde l é o comprimento de um vetor binário $x_g = x_g^1, x_g^2, \dots, x_g^l$ que representa uma solução no espaço de busca [44]. Apesar de ser o tipo de representação mais simples, em muitas aplicações a limitação do domínio implica na diminuição do poder de expressão da codificação binária [14].

Uma alternativa frequentemente utilizada é a codificação real. Esse tipo de codificação é usualmente empregada quando os alelos são distribuídos em um intervalo contínuo. Dessa maneira, o genótipo é definido como:

$$\phi_g = \mathbb{R}^l, \quad (2-2)$$

onde l é o comprimento de um cromossomo real valorado. A utilização da codificação real geralmente implica no uso de estratégias evolutivas baseadas em modelos probabilísticos, como os Algoritmos de Estimação de Distribuição [42].

Como dito anteriormente, a representação de uma solução consiste na definição do genótipo e do mapeamento do genótipo em fenótipo. Portanto, uma determinada função objetivo f pode ser decomposta em duas outras funções: f_g que mapeia o genótipo ϕ_g em fenótipo ϕ_f e f_f que mapeia o fenótipo ϕ_f em um valor no plano real \mathbb{R} .

$$\begin{aligned} f_g(x_g) : \phi_g &\rightarrow \phi_f \\ f_f(x_f) : \phi_f &\rightarrow \mathbb{R}, \end{aligned} \quad (2-3)$$

sendo $f = f_f(f_g(x_g))$. O mapeamento do genótipo em fenótipo f_g é determinado pelo tipo de codificação utilizado. f_f é a função que atribui um valor real a cada solução $x_f \in \phi_f$ [44]. Dessa maneira, operadores de busca atuam no nível do genótipo enquanto a avaliação da qualidade das soluções é realizada no nível do fenótipo [16].

Uma importante etapa da implementação do AG é a inicialização da população. Geralmente, a inicialização é feita de maneira aleatória, com o objetivo de proporcionar maior diversidade à população. No entanto, o conhecimento a priori da distribuição das melhores soluções pode ser útil nesse processo. Neste caso, o AG poderá ganhar em acurácia e eficiência computacional [16].

2.2.2 Seleção

A seleção é a etapa responsável pela escolha de dois indivíduos para a etapa de cruzamento. Tal seleção é feita com base na aptidão de cada indivíduo, calculada por meio de uma função objetivo f . Em problemas de otimização, a aptidão equivale ao custo da solução. Indivíduos com maiores níveis de aptidão possuem maiores probabilidades de serem selecionados [19, 34]. Se o problema for de minimização, as soluções mais aptas são as de menor custo. Os métodos de seleção mais utilizados são:

- **Ranking:** os indivíduos são organizados de acordo com os níveis de aptidão. A probabilidade do indivíduo ser selecionado depende da posição que ele ocupa no *ranking*;
- **Roleta:** a roleta é formada pelo somatório da aptidão de cada indivíduo. A tamanho da área destinada a um indivíduo na roleta é proporcional ao seu nível de aptidão. A seleção é feita por um sorteio, onde os indivíduos com maiores níveis de aptidão possuem maiores chances de serem selecionados;
- **Torneio:** nessa estratégia são realizados t torneios, sendo t o tamanho da população. Para cada torneio são sorteados j indivíduos que competem entre si. O indivíduo com melhor desempenho na função objetivo é selecionado;
- **Uniforme:** diferente dos outros métodos, na seleção uniforme todos os indivíduos possuem a mesma probabilidade de serem selecionados. O ponto negativo deste método é sua lenta taxa de convergência.

2.2.3 Cruzamento

O cruzamento é o processo que recombina indivíduos previamente selecionados com o objetivo de gerar descendentes mais aptos no contexto do problema considerado. Tal processo é fundamental na diversificação da população e na manutenção das características dos melhores indivíduos [34].

Algumas das principais estratégias de cruzamento são apresentadas a seguir.

- **Cruzamento de um ponto:** o Algoritmo Genético clássico utiliza este tipo de cruzamento, onde dois indivíduos, previamente selecionados, são repartidos em um ponto aleatoriamente escolhido. A primeira e a segunda parte dos indivíduos pai_1 e pai_2 , respectivamente, são combinadas para formar o filho $filho_1$. Em seguida, a segunda e a primeira parte dos indivíduos pai_1 e pai_2 , respectivamente, são combinadas para formar o filho $filho_2$. Ambos fazem parte da nova geração e serão adicionados à população.

- **Cruzamento multiponto:** o funcionamento do cruzamento multiponto é semelhante ao de um ponto. A diferença é que mais de um ponto de cruzamento pode ser selecionados;
- **Cruzamento de dois pontos:** nesta estratégia são selecionados dois pontos de cruzamento. Os alelos localizados entre os pontos de cruzamento em cada indivíduo pai são intercambiados, gerando dois indivíduos filhos;
- **Cruzamento uniforme:** o funcionamento do cruzamento uniforme é bastante diferente dos demais, pois não utiliza pontos de cruzamento. Uma máscara com o mesmo comprimento dos pais é gerada contendo 1s e 0s, distribuídos de maneira aleatória. Se o alelo da máscara for igual a 1, o alelo do indivíduo pai_1 será copiado para o $filho_1$. Se for igual a 0, o alelo a ser copiado será proveniente do pai_2 . O mesmo processo é utilizado para gerar o $filho_2$, mas com a máscara invertida. Onde 0 passa a valer 1 e vice-versa. A cada cruzamento uma nova máscara é gerada.

Após a introdução da estratégia de cruzamento de um ponto, muitas das técnicas apresentadas envolve m mais de um ponto de cruzamento. A adição de mais pontos de cruzamento proporciona uma exploração mais minuciosa no espaço de busca. Contudo, quanto mais pontos de cruzamento, maiores a probabilidades de rompimento dos *building blocks* [16].

2.2.4 Mutação

O principal responsável pela variabilidade do AG é a mutação, processo pelo qual o indivíduo fica sujeito a sofrer alterações em sua estrutura. No Algoritmo Genético clássico, cada alelo do indivíduo possui uma pequena probabilidade de ser alterado. Normalmente, tal probabilidade é igual a $1/l$. Outros valores podem ser utilizados, porém, uma probabilidade muito alta pode fazer com que a população perca suas características fundamentais [34].

2.3 Hipótese dos *Building Blocks*

2.3.1 Teorema dos Esquemas

A teoria original do Algoritmo Genético fundamenta-se na representação binária de soluções e na noção de esquemas [35]. Esquema é um padrão genético capaz de descrever um conjunto de indivíduos compostos por uma quantidade de genes fixos. Mais especificamente, esquemas são sequências de genes representados por caracteres pertencentes ao alfabeto $\sigma = \{0, 1, *\}$, onde o caractere coringa (*) pode simbolizar

quaisquer dos outros caracteres de σ . Por exemplo, o esquema $\mathbf{h} = 1****$ é capaz de produzir qualquer indivíduo, de comprimento $l = 5$, cujo primeiro alelo é igual a 1.

No Teorema dos Esquemas, apresentado por Holland em 1975 [27], existem duas características importantes que devem ser levadas em consideração: *i*) a ordem $O(\mathbf{h})$ de um esquema é definida como o número de alelos fixos no cromossomo, ou seja, a quantidade de alelos diferentes de *; e *ii*) o comprimento definitório $\delta(\mathbf{h})$ do esquema é a distância entre os alelos fixos mais distantes. Por exemplo, considere os esquemas \mathbf{h}_1 e \mathbf{h}_2 , onde \mathbf{h}_1 possui $O(\mathbf{h}) = 3$ e $\delta(\mathbf{h}) = 4$ e \mathbf{h}_2 possui $O(\mathbf{h}) = 2$ e $\delta(\mathbf{h}) = 3$.

$$\mathbf{h}_1 = 1*0*1,$$

$$\mathbf{h}_2 = *1**0.$$

Segundo Holland, ao avaliar um indivíduo, o AG está, implicitamente, avaliando todos os esquemas relacionados a ele. Dessa maneira, é possível determinar a aptidão do esquema \mathbf{h} a partir do aptidão de todos os indivíduos relacionados a tal esquema e presentes na população, como mostra a Equação (2-4).

$$f(\mathbf{h}) = \frac{1}{|\mathbf{h}|} \sum_{x \in \mathbf{h}} f(x), \quad (2-4)$$

onde $|\mathbf{h}|$ é o número de indivíduos \mathbf{x} que são instâncias do esquema \mathbf{h} .

Descrito pela Equação (2-5), o Teorema dos Esquemas explica o aumento da quantidade de instâncias de um esquema a cada nova geração t .

$$r(\mathbf{h}, P)_{ger+1} \geq r(\mathbf{h}, P)_{ger} \frac{\bar{f}(\mathbf{h})_{ger}}{\bar{f}_{ger}} (1 - x_c \frac{\delta(\mathbf{h})}{l-1}) (1 - p_m)^{O(\mathbf{h})}, \quad (2-5)$$

onde

- $r(\mathbf{h}, P)_{ger}$ é a quantidade de indivíduos relacionados ao esquema \mathbf{h} na população P na geração ger ;
- $\bar{f}(\mathbf{h})_{ger}$ é a média do nível de aptidão dos indivíduos relacionados ao esquema \mathbf{h} , ou, simplesmente, a aptidão do esquema;
- \bar{f}_{ger} é a média da aptidão de todos os indivíduos da população na geração ger ;
- $(1 - x_c \frac{\delta(\mathbf{h})}{l-1})$ representa a probabilidade de o esquema \mathbf{h} sobreviver ao processo de cruzamento;
- $(1 - p_m)^{O(\mathbf{h})}$ representa a probabilidade de o esquema \mathbf{h} sobreviver ao processo de mutação.

De maneira geral, o teorema dos esquemas consiste do produto das etapas de seleção, cruzamento e mutação, definidos pelos termos $r(\mathbf{h}, P)_{ger} \frac{\bar{f}(\mathbf{h})_{ger}}{\bar{f}_{ger}}$, $(1 - x_c \frac{\delta(\mathbf{h})}{l-1})$ e $(1 - p_m)^{O(\mathbf{h})}$, respectivamente. Quando a aptidão do esquema é maior que a aptidão de

toda a população ($f(\mathbf{h})_{ger} > \bar{f}_{ger}$), a etapa de seleção tende a proporcionar o aumento da quantidade de indivíduos relacionados ao esquema avaliado. No entanto, para que isso realmente ocorra, o comprimento definitório do esquema ($\delta(\mathbf{h})$) deve ser curto o suficiente para evitar a ruptura do esquema durante a etapa de cruzamento. De maneira semelhante, a ordem do esquema $O(\mathbf{h})$ deve ser baixa para evitar o rompimento do esquema durante o processo de mutação. Assim, problemas com esquemas com aptidão elevada, comprimento definitório curto e baixa ordem podem ser facilmente solucionados utilizando AG [43].

2.3.2 *Building Blocks*

O termo *building block* (BB) foi informalmente citado pela primeira vez por Holland [27] na formulação do Teorema dos Esquemas. Goldberg [21], em 1987, definiu o BB como sendo um esquema com aptidão acima da média, comprimento definitório curto e baixa ordem. A Hipótese do *Building Block*, também descrita por Goldberg [21], diz que os BBs são avaliados, selecionados e recombinados de modo a gerar novas e melhores soluções a cada geração. Basicamente, tal hipótese diz que se o problema puder ser dividido em subproblemas, estes serão solucionados separadamente e então reagrupados para formar boas soluções. Em outras palavras, um BB pode ser descrito como uma solução para um subproblema representado por um esquema [43]. Relembrando, um esquema com aptidão elevada, comprimento definitório curto e baixa ordem tende a ser propagado ao longo das gerações, aumentando o número de indivíduos relacionados a ele. Contudo, se o esquema possui comprimento definitório longo e/ou alta ordem, ele é frequentemente rompido pelos operadores de cruzamento e mutação. Consequentemente, tal esquema não será propagado ao longo das gerações, prejudicando o desempenho do AG.

Por exemplo, considere os dois esquemas \mathbf{h}_1 e \mathbf{h}_2 :

$$\mathbf{h}_1 = 11*****,$$

$$\mathbf{h}_2 = 1****111.$$

Em qualquer esquema de comprimento l existem $l - 1$ possíveis pontos de cruzamento e $O(\mathbf{h})$ candidatos ao processo de mutação. No caso de \mathbf{h}_1 , a probabilidade de rompimento do esquema durante o cruzamento é de apenas $\frac{1}{l-1}$, uma vez que apenas 1 dos $l - 1$ possíveis pontos de cruzamento causaria o rompimento dos BBs. Por outro lado, considerando \mathbf{h}_2 , a probabilidade de rompimento do esquema é de $\frac{l-1}{l-1}$, ou 1, já que existem $l - 1$ pontos de cruzamento separando os alelos que formam o *building block*. Considerando agora a etapa de mutação, o esquema \mathbf{h}_1 possui 2 pontos de rompimento ($O(\mathbf{h}_1) = 2$) do BB enquanto \mathbf{h}_2 possui 4 ($O(\mathbf{h}_2) = 4$). Dessa maneira, BBs formados por

poucos alelos fixos e próximos podem ser eficientemente processados por AGs utilizando operadores de cruzamento de um ponto e mutação, enquanto BBs compostos por muitos alelos e distantes possuem poucas chances de sobreviverem a tal processo [48].

Como destacado por Mitchell *et al.* [34], o AG funciona muito bem quando esquemas com elevado nível de aptidão, comprimento definitório curto e de baixa ordem são recombinaados para formar esquemas com aptidão ainda mais elevada. É o que ocorre no caso de esquemas como \mathbf{h}_1 . Por outro lado, considerando o comprimento definitório do esquema \mathbf{h}_2 e sua alta ordem, dificilmente os BBs de tal esquema sobreviverão aos sucessivos processos de cruzamento e mutação. Neste caso, ocorre o chamado rompimento do *building block*.

Uma alternativa ao AG, chamada de Algoritmo Genético Compacto, foi proposta por Harik *et al.* [26] e processa os *building blocks* de maneira independente. O algoritmo citado é descrito na próxima Seção.

2.4 Algoritmo Genético Compacto

O Algoritmo Genético Compacto (AGC), introduzido por Harik *et al.* [26], é uma estratégia evolutiva que imita o comportamento de ordem um do Algoritmo Genético clássico com operador de cruzamento de um ponto. Porém, diferentemente da versão clássica do AG que adota uma população inicial com codificação binária, o AGC utiliza apenas um vetor de probabilidades com codificação real para representar toda a população. Tal abordagem permite que cada indivíduo seja avaliado no contexto de cada alelo. O AG clássico, por outro lado, avalia o indivíduo como um todo, o que pode levar o algoritmo a tomar decisões erradas. Adicionalmente, o AGC adota a probabilidade para gerar novos indivíduos e direcionar a busca, substituindo os operadores de cruzamento e mutação presentes no AG. A Relação (2-6) mostra como o vetor de probabilidades é representado.

$$\mathbf{p} = [p_1 \quad p_2 \quad \dots \quad p_l]^T \quad (2-6)$$

O vetor de probabilidades \mathbf{p} tem comprimento l e todos os seus elementos são, inicialmente, iguais a 0,5 ($\mathbf{p} = [0,5 \quad 0,5 \quad \dots \quad 0,5]^T$). A cada geração, dois indivíduos (*novo_ind_a* e *novo_ind_b*) são aleatoriamente gerados a partir de \mathbf{p} . Em seguida os indivíduos competem entre si em uma determinada função objetivo. O indivíduo com melhor desempenho é então utilizado para atualizar o vetor \mathbf{p} . A etapa de atualização é

realizada da seguinte maneira conforme descrito na Equação (2-7).

$$\mathbf{p}(i) = \begin{cases} \mathbf{p}(i) + 1/n, & \text{se } vencedor(i) = 1 \text{ e } vencedor(i) \neq perdedor(i), \\ \mathbf{p}(i) - 1/n, & \text{se } vencedor(i) = 0 \text{ e } vencedor(i) \neq perdedor(i), \\ \mathbf{p}(i), & \text{if } vencedor(i) = perdedor(i), \end{cases} \quad (2-7)$$

onde n é o tamanho da população simulada e $0 \leq i < l$.

O Algoritmo 2.2 mostra o pseudocódigo do AGC.

Algoritmo 2.2: Algoritmo Genético Compacto, adaptado de Harik *et al.* [26].

1. Seja l a quantidade de alelos do indivíduo
 2. $t \leftarrow 0$;
 3. **Enquanto** $t < l$
 4. $p(t) \leftarrow 0.5$; //Inicialização do vetor de probabilidades $t \leftarrow t + 1$;
 5. **Fim Enquanto**
 6. $t \leftarrow 0$;
 7. **Enquanto** $t < \text{numero_maximo_de_geracoes}$ ou $p_nao_tenha_convergido$
 8. **novo_ind_a** \leftarrow *gera_individuo*(\mathbf{p});
 9. **novo_ind_b** \leftarrow *gera_individuo*(\mathbf{p});
 10. $vencedor, perdedor \leftarrow \text{compete}(\text{novo_ind_a}, \text{novo_ind_b})$;
 11. $\mathbf{p} \leftarrow \text{atualiza_p}(vencedor, perdedor)$; //Atualiza o vetor de probabilidades na direção do vencedor.
 12. **Fim Enquanto**
-

O trecho do código entre as linhas 2 e 5 servem apenas para inicializar o vetor de probabilidades. A partir da linha 7 o algoritmo começa a busca pela melhor solução. Dois indivíduos são gerados a partir de \mathbf{p} nas linhas 8 e 9. Tais indivíduos competem entre si em uma determinada função na linha 10. O vetor de probabilidades é atualizado, na linha 11, na direção do indivíduo vencedor. À medida que o algoritmo evolui, o vetor de probabilidades \mathbf{p} tende a convergir para uma solução explícita [25]. O algoritmo interrompe sua execução quando todos os valores do vetor são iguais a 0 ou a 1, ou quando atingir o número máximo de gerações.

A geração de indivíduos no AGC é equivalente a executar uma infinidade de mutações a cada geração no AG clássico. Dessa maneira, o AGC descorrelaciona completamente os alelos, tratandos-os individualmente. Por exemplo, considere os pares de esquemas $vencedor_1$ e $perdedor_1$ e $vencedor_2$ e $perdedor_2$:

$$\begin{aligned} vencedor_1 &= 11*****, perdedor_1 = 00***** \\ vencedor_2 &= 1*****1, perdedor_2 = 0*****0 \end{aligned}$$

Considere que o esquema $vencedor_1$ obteve melhor desempenho que o esquema $perdedor_1$ em uma determinada função objetivo. Então o vetor de probabilidades \mathbf{p} será atualizado na direção do $vencedor_1$. Dessa maneira, haverá maior probabilidade de preservação do *building block* formado pelos dois primeiros genes na próxima geração. Comportamento semelhante é observado quando considerado o par de esquemas $vencedor_2$ e $perdedor_2$. Portanto, não há que se falar de rompimento de *building block* no AGC. No entanto, também não há garantia de que BBs serão propagados nas gerações seguintes, uma vez que existe apenas uma maior probabilidade de estarem presentes nos próximos indivíduos gerados.

2.5 Otimização Multiobjetivo

Um problema de Otimização Multiobjetivo consiste na minimização (ou maximização) de vários objetivos, geralmente conflitantes, simultaneamente. Esta característica impossibilita, muitas vezes, a determinação de uma única solução capaz de otimizar todos os objetivos. Por isso, solucionar problemas desta natureza significa obter um conjunto de soluções que satisfaça as restrições impostas e otimize as diversas funções objetivo. Por exemplo, considere o caso de desenvolvimento de um *hardware* complexo. Idealmente, espera-se que o *hardware* desenvolvido associe o menor custo à máxima performance. Contudo, tais objetivos são claramente conflitantes. Normalmente, a arquitetura de menor custo não entrega a melhor performance. Da mesma maneira, o *hardware* com melhor performance dificilmente será o de menor custo [54]. A Figura 2.2 ilustra o exemplo analisado.

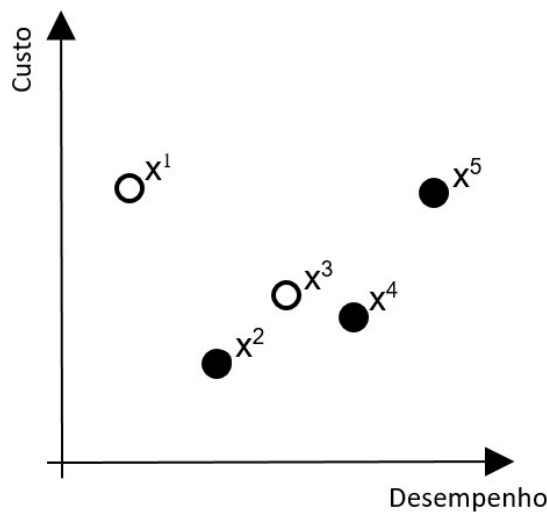


Figura 2.2: Soluções para o problema de minimização do custo do hardware e maximização de seu desempenho.

No caso do exemplo analisado, cada bola representa uma solução para o problema considerado. Como pode ser observado, não existe uma solução ótima, mas um conjunto de soluções intermediárias, formado pelas soluções $\mathbf{x}^1, \mathbf{x}^2$ e \mathbf{x}^3 que podem ser classificadas de modo a auxiliar na tomada de decisão. Adicionalmente, é possível notar que o problema é maximizar o desempenho do *hardware* e, ao mesmo tempo, minimizar seu custo de produção. Neste caso, é necessário converter uma das funções objetivo de modo que todas sejam de minimização ou de maximização [54].

Formalmente, o problema de Otimização Multiobjetivo pode ser descrito pela Equação (2-8) [53]:

$$\begin{aligned} & \text{Minimizar/Maximizar} \quad f_i(\mathbf{x}) \quad i = 1, 2, \dots, l \\ & \text{s.a.} \quad g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J \\ & \quad \quad h_k(\mathbf{x}) = 0, \quad k = 1, 2, \dots, K \end{aligned} \quad (2-8)$$

onde $f_i(\mathbf{x})$ é a i -ésima função objetivo, g_j e h_k representam restrições do problema e \mathbf{x} é um vetor composto pelas s variáveis de decisão e representa uma solução no espaço de busca. Uma solução \mathbf{x} só será factível se atender a todas as restrições impostas pela problema g_j e h_k [13].

Como dito anteriormente, a maioria dos problemas de tomada de decisão envolvem a otimização de múltiplos objetivos. Em princípio, a otimização multiobjetivo é bastante diferente da otimização mono-objetivo. Na otimização mono-objetivo a finalidade é determinar a melhor solução, que pode ser global máximo ou global mínimo dependendo do problema ser de maximização ou minimização. No caso de otimização multiobjetivo, pode não existir uma solução que seja melhor em todos os objetivos. Tipicamente, existe um conjunto de soluções que são melhores que as demais soluções contidas no espaço de busca e que podem ser classificadas de acordo com o conceito de Dominância de Pareto [47]. Tal conceito é descrito a seguir.

2.5.1 Dominância de Pareto

Em problemas de otimização multiobjetivo, geralmente, são consideradas todas as soluções que são melhores que as demais soluções factíveis do espaço de busca, mas são piores que outras soluções em um ou mais objetivos. Tais soluções são chamadas de soluções não dominadas, ou Pareto-ótimo, enquanto as demais soluções são chamadas de soluções dominadas. Para classificar as soluções como dominadas ou não-dominadas, é preciso entender o conceito de Dominância de Pareto.

Considere um problema de minimização multiobjetivo e que os vetores x^1 e x^2 representem soluções para o problema. Então a solução x^1 domina a solução x^2 ($x^1 \preceq x^2$) se e somente se [47]:

- x^2 não é melhor que x^1 em nenhum dos objetivos; e
- x^1 é estritamente melhor que x^2 em pelo menos um objetivo

Dessa maneira, x^1 é considerada uma solução Pareto-ótima, ou dominante, enquanto x^2 é uma solução dominada. Se não for possível estabelecer relação de dominância entre duas soluções, então essas são ditas indiferentes.

É possível ainda classificar a relação de dominância como forte ou fraca. A solução x^1 domina fortemente a solução x^2 ($x^1 \prec x^2$) se x^1 é estritamente melhor que x^2 em todos os n objetivos analisados. Por outro lado, diz-se que x^1 domina fracamente a solução x^2 se x^1 não domina fortemente a solução x^2 , ou seja, x^1 não é melhor que x^2 em todos os s objetivos [14].

A relação de dominância deve também satisfazer as seguintes propriedades:

- Não reflexiva - Uma solução não pode ser dominada por ela mesma.
- Não simétrica - $x^1 \preceq x^2$ não implica $x^2 \preceq x^1$
- Transitiva - Se $x^1 \preceq x^2$ e $x^2 \preceq x^3$, então $x^1 \preceq x^3$

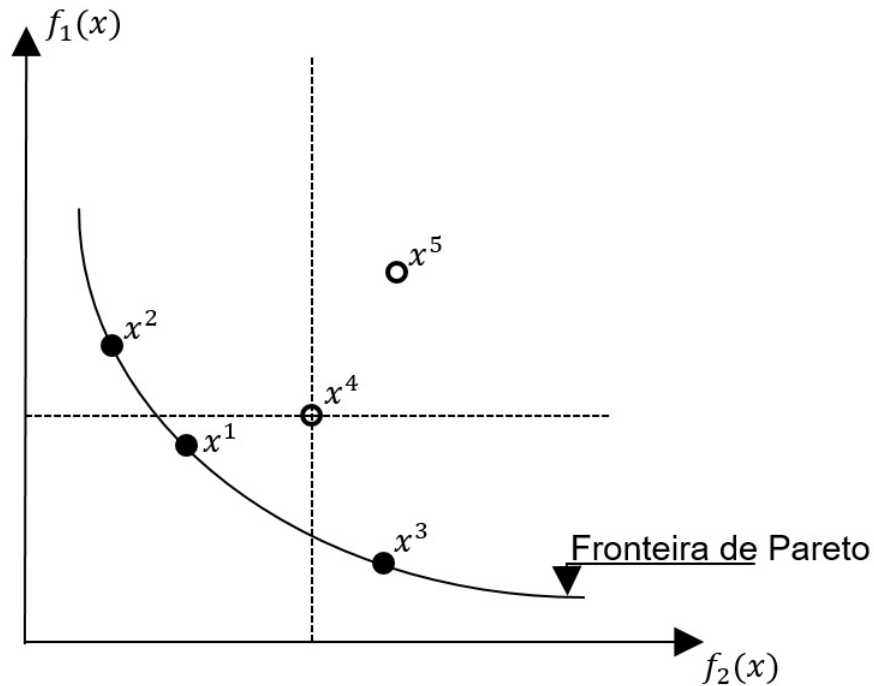


Figura 2.3: Ilustração do conceito de soluções Pareto-ótimas.

A Figura 2.3 ilustra as definições apresentadas para um problema de minimização multiobjetivo. Ante o exposto e observando a Figura 2.3, nota-se que a solução x^1 domina fortemente a solução x^4 , que por sua vez domina fortemente a solução x^5 . Logo, pela propriedade de transitividade, x^1 domina fortemente a solução x^5 . Portanto, x^4 e x^5 são soluções fortemente dominadas por x^1 . Nota-se ainda que as soluções x^2 e x^3 dominam

fracamente x^4 . Dada a relação existente entre x^4 e x^5 , conclui-se que x^2 e x^3 dominam fortemente x^5 . Por outro lado, as soluções x^1 , x^2 e x^3 não são dominadas por nenhuma outra solução, sendo, portanto, consideradas soluções Pareto-ótimas. Considerando a totalidade de soluções contidas no espaço de busca, o conjunto de soluções não-dominadas forma a chamada Fronteira de Pareto.

Em geral, algoritmos de otimização multiobjetivo devem concluir duas metas principais:

- Obter um conjunto de soluções que esteja o mais próximo possível da fronteira de Pareto;
- Encontrar um conjunto de soluções com maior diversidade possível.

Como já destacado, em problemas que buscam a otimização de objetivos conflitantes, geralmente, não existe uma única solução que seja ótima com respeito a todos os objetivos. Por isso, observando cuidadosamente a Figura 2.3, é possível perceber que cada solução Pareto-ótima pode ser classificada de acordo com seu nível de importância para cada objetivo otimizado. Fica claro, por exemplo, que x^3 é a melhor solução para f_1 e que x^2 representa a melhor solução para a função f_2 . Em situações como esta, utilizam-se técnicas baseadas no conhecimento sobre o problema para selecionar a solução mais adequada.

Os algoritmos propostos neste trabalho não são capazes de determinar a Fronteira de Pareto em uma única execução. Logo, não podem ser considerados algoritmos de otimização multiobjetivo. Contudo, é possível encontrar soluções pertencentes à Fronteira de Pareto utilizando o conceito de ponderação dos objetivos. Para melhor elucidar essa questão, o método da Soma Ponderada é descrito a seguir.

2.5.2 Método da Soma Ponderada

Uma característica inerente aos problemas de otimização multiobjetivo é a dificuldade em determinar uma única solução que otimize todos os objetivos avaliados. Uma alternativa para superar tal dificuldade é entender as prioridades do tomador de decisão. O tomador de decisão é aquele que, dotado de conhecimento prévio, interfere no processo de solução definindo quais funções objetivo possuem maior relevância para o problema. Neste cenário, as principais técnicas para a solução de problemas de otimização multiobjetivo podem ser classificadas de acordo com o momento em que o tomador de decisão interfere no processo [8].

Na estratégia Baseada em Preferência, o tomador de decisão interfere no processo de otimização antes do seu início. Neste tipo de abordagem, o tomador de decisão precisa determinar bons conhecimentos a respeito das funções objetivo e das restrições impostas pelo problema. A partir de então, é possível estabelecer prioridades entre os objetivos

e, de alguma forma, agregá-los em uma única função. Ao final do processo de otimização, uma única solução é encontrada [3, 17].

Dentre as técnicas Baseadas em Preferência, a Soma Ponderada é provavelmente a mais utilizada. Baseada em um conceito bastante intuitivo, a Soma Ponderada escala um conjunto de objetivos em uma única função multiplicando cada objetivo por um coeficiente de ponderação. Embora a ideia seja bastante simples, uma importante questão é saber quais valores atribuir a cada coeficiente. Usualmente, tais valores são proporcionais à importância de cada objetivo no contexto do problema. Por exemplo, o custo de produção de um *hardware* pode ser mais importante que seu desempenho. Neste caso, a minimização do custo de produção tem maior prioridade. Apesar de existirem maneiras de quantificar os coeficientes de ponderação a partir desta informação qualitativa, a Soma Ponderada requer um valor preciso para ponderar cada objetivo [14].

Determinar o valor apropriado para os coeficientes de ponderação também depende da escala de cada função objetivo. É provável que diferentes objetivos tenham diferentes magnitudes. No exemplo anterior, o custo de produção de um *hardware* pode estar na casa dos milhares de reais enquanto seu desempenho poder ser da ordem gigabits. Por isso, antes de ponderar os objetivos, é recomendável que todos tenham aproximadamente a mesma magnitude [14].

Após a normalização dos objetivos, uma função F poder ser formulada para somar os objetivos ponderados. Dessa maneira, o problema de otimização multiobjetivo é convertido em um problema de otimização mono-objetivo, como apresentado pela Equação (2-9).

$$\text{Minimizar/Maximizar } F = \sum_{i=1}^n w_i f_i(x) \quad (2-9)$$

onde w_i é o coeficiente de ponderação associado à função objetivo f_i e determina a prioridade associada ao objetivo. w

Considerando que F seja a combinação linear de duas funções objetivo, f_1 e f_2 , então as soluções para o problema podem ser representadas por retas traçadas no espaço de busca. Isso acontece porque todas as soluções pertencentes a uma reta possuem o mesmo valor de F . A Figura 2.4 mostra várias retas traçadas no espaço de busca de um problema de minimização de dois objetivos [14].

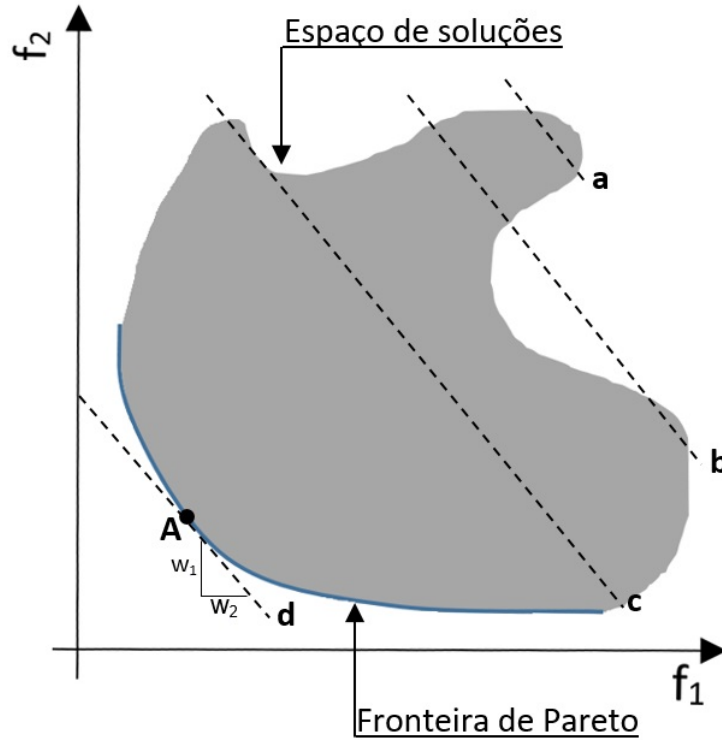


Figura 2.4: Representação do espaço de busca de um problema de minimização com dois objetivos.

A intersecção das retas a , b , c e d com o espaço de busca representam possíveis soluções para o problema considerado, onde a posição de cada reta depende do valor de F . Por se tratar de um problema de minimização, fica claro que a melhor solução é representada pela intersecção entre reta d e o espaço de busca. Neste caso, o ponto A representa uma solução pertencente à Fronteira de Pareto. A inclinação da reta, por sua vez, depende dos valores dos coeficientes de ponderação, w_1 e w_2 . Tais coeficientes podem assumir quaisquer valores, contudo, é usual que os pesos obedeçam às seguintes restrições [3]:

$$\begin{aligned} s.a. \quad & w_i \neq 0 \\ & \sum_{i=1}^n w_i = 1 \end{aligned} \quad (2-10)$$

Dessa maneira, conforme a Equação (2-10), a inclinação da reta é dada pela razão $-w_1/w_2$.

A técnica de Soma Ponderada é utilizada principalmente na solução de problemas em que o espaço de busca é convexo, como ilustrado no exemplo anterior. Neste caso, a Soma Ponderada garante que a solução encontrada sempre pertencerá à Fronteira de Pareto [14].

Embora a ideia do método seja bastante simples, é possível elencar uma série de desvantagens relacionadas à Soma Ponderada. Uma delas é a dificuldade em estabelecer os coeficientes de ponderação que correspondam aos critérios estabelecidos pelo tomador

de decisão. Adicionalmente, dada a sensibilidade do método à variação nos pesos, qualquer mudança pode alterar significativamente a solução encontrada. Uma vez que o resultado encontrado não é satisfatório, o processo de otimização deverá ser reiniciado utilizando novos coeficientes de ponderação. Tal característica torna a solução ótima totalmente subjetiva, a critério do tomador de decisão [3].

A Soma Ponderada também não é o método mais indicado quando o problema envolve a otimização de funções não-lineares. Quando isso ocorre, mesmo que os coeficientes de ponderação estejam normalmente distribuídos, não há garantia de obtenção de soluções normalmente distribuídas na Fronteira de Pareto [14]. Contudo, a principal limitação da Soma Ponderada surge quando o espaço de busca do problema de otimização considerado é não convexo, como ilustrado na Figura 2.5.

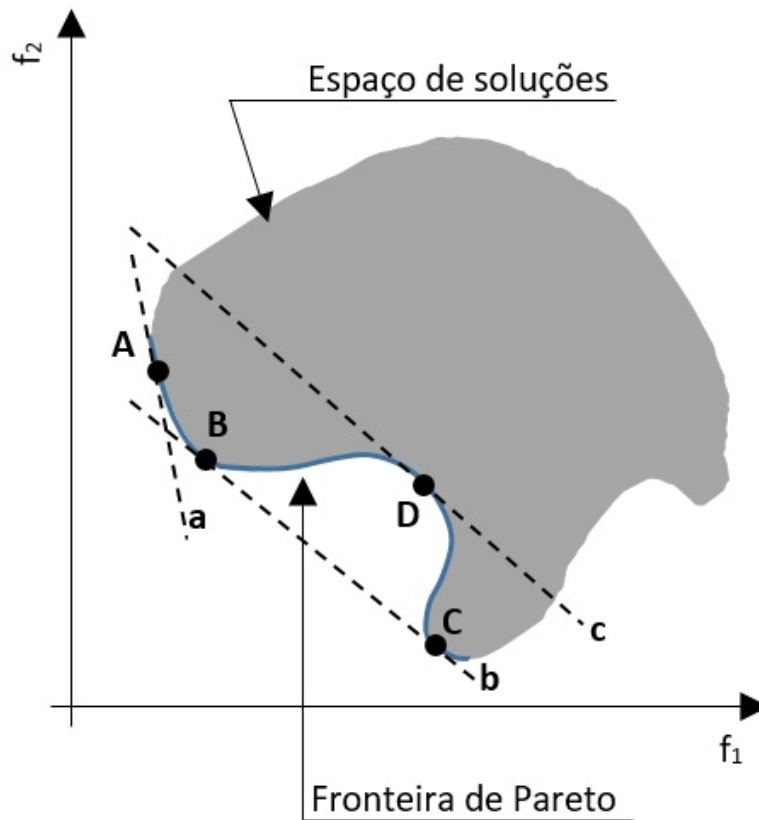


Figura 2.5: Representação do espaço de busca não convexo.

Em problemas de otimização como o ilustrado pela Figura 2.5, as soluções posicionadas na Fronteira de Pareto entre as soluções B e C não são consideradas Pareto-ótimas. Isso porque os valores de F referentes às soluções encontradas nesse intervalo sempre serão piores que os valores de F relativos às soluções presentes em outras regiões da Fronteira de Pareto [14]. É importante ressaltar ainda que apenas um valor de F é determinado ao final de cada execução do método. Para que múltiplas soluções sejam encontradas, o método deve ser executado várias vezes com diferentes valores atribuídos aos coeficientes de ponderação.

Implementações Propostas

Como descrito no Capítulo 2, o Algoritmo Genético (AG) possui boas chances de encontrar as melhores soluções quando estas são representadas por esquemas de baixa ordem. Por outro lado, quando os *building blocks* são formados por alelos distantes, os operadores de recombinação tendem a prejudicar o desempenho do AG. Neste caso, o Algoritmo Genético Compacto (AGC) representa uma alternativa ao AG visto que sua formulação utiliza uma abordagem diferente para tratamento dos *building blocks*.

Originalmente introduzido por Harik *et al.*[26], o AGC imita o comportamento do AG com cruzamento de um ponto quando as melhores soluções podem ser representadas por esquemas de ordem um. Portanto, no melhor caso, o AG possui desempenho compatível com o AGC. Ao utilizar um vetor de probabilidades para gerar novas soluções, o AGC suprime as etapas de cruzamento e mutação proporcionando melhores chances de sobrevivência aos *building blocks*.

Como já destacado, a otimização multiobjetivo tem sido largamente utilizada na solução de problemas de seleção de variáveis. No sentido de proporcionar uma nova ferramenta para esse tipo de problema, este trabalho apresenta e compara três versões do Algoritmo Genético Compacto com avaliação multiobjetivo. A primeira versão utiliza o conceito de dominância de Pareto para atualizar o vetor de probabilidades. A segunda versão, baseada na primeira, adiciona a capacidade de estabelecer prioridades entre os objetivos avaliados. A última versão implementada utiliza a soma ponderada para transformar os vários objetivos em uma única função. Todas as versões do Algoritmo Genético Compacto com avaliação multiobjetivo são apresentadas nas seções a seguir.

3.1 Algoritmo Genético Compacto com Dominância

A primeira versão multiobjetivo do AGC, apresentada pelo Algoritmo 3.1, é denominada Algoritmo Genético Compacto com Dominância (AGC-D).

Algoritmo 3.1: Algoritmo Genético Compacto com Dominância - AGC-D

-
1. Seja l o número de variáveis disponíveis.
 2. $t \leftarrow 0$;
 3. **Enquanto** $t < l$
 4. $p(t) \leftarrow 0,5$; //Inicialização do vetor de probabilidades $t \leftarrow t + 1$;
 5. **Fim enquanto**
 6. $t \leftarrow 0$;
 7. **Enquanto** ($t < \text{número máximo de gerações}$ ou p não tenha convergido)
 8. **novo_ind_a** $\leftarrow \text{gera_individo}(\mathbf{p})$; //gera indivíduo a partir de \mathbf{p}
 9. **novo_ind_b** $\leftarrow \text{gera_individo}(\mathbf{p})$; //gera indivíduo a partir de \mathbf{p}
 10. $[\text{vencedor}_{f_1}, \text{perdedor}_{f_1}] \leftarrow \text{compete}_{f_1}(\text{novo_ind_a}, \text{novo_ind_b})$
 11. $\mathbf{p} \leftarrow \text{atualiza_p}(\text{vencedor}_{f_1}, \text{perdedor}_{f_1})$ //Atualiza o vetor de probabilidades na direção do vencedor.
 12. $[\text{vencedor}_{f_2}, \text{perdedor}_{f_2}] \leftarrow \text{compete}_{f_2}(\text{novo_ind_a}, \text{novo_ind_b})$
 13. $\mathbf{p} \leftarrow \text{atualiza_p}(\text{vencedor}_{f_2}, \text{perdedor}_{f_2})$ //Atualiza o vetor de probabilidades na direção do vencedor.
 14. **Fim Enquanto**
-

Considerando l variáveis disponíveis, o algoritmo inicializa todos os alelos do vetor de probabilidades (\mathbf{p}) com 0,5. Nesta etapa, cada variável possui a mesma probabilidade de ser selecionada. Nos passos 9 e 10, a partir de \mathbf{p} , são gerados dois indivíduos para serem avaliados pela primeira função objetivo no passo 11. O indivíduo com maior aptidão é definido como vencedor e o pior como perdedor. No passo 12 o vetor de probabilidades é atualizado no sentido do indivíduo vencedor e contrário ao perdedor conforme Relação (3-1). Nos passos seguintes, os indivíduos são avaliados pela segunda função objetivo e o vetor de probabilidades \mathbf{p} é atualizado novamente seguindo a mesma estratégia do passo 12. As etapas compreendidas entres os passos 9 e 14 são repetidas até que o número máximo de gerações tenha sido alcançado ou até que \mathbf{p} tenha convergido para uma solução explícita, ou seja, todos os *bits* de \mathbf{p} sejam iguais a 0 ou 1. Dessa maneira, o vetor de probabilidades gera soluções considerando todos os objetivos avaliados.

Para melhor entendimento do funcionamento do passo de atualização no contexto da dominância de Pareto, faz-se necessária a reinserção da equação que descreve tal passo.

$$\mathbf{p}(i) = \begin{cases} \mathbf{p}(i) + 1/n, & \text{se } \text{vencedor}(i) = 1 \text{ e } \text{vencedor}(i) \neq \text{perdedor}(i), \\ \mathbf{p}(i) - 1/n, & \text{se } \text{vencedor}(i) = 0 \text{ e } \text{vencedor}(i) \neq \text{perdedor}(i), \\ \mathbf{p}(i), & \text{se } \text{vencedor}(i) = \text{perdedor}(i), \end{cases} \quad (3-1)$$

onde n é o tamanho da população simulada e $0 \leq i < l$.

A partir da Equação (3-1) é possível observar que o i -ésimo *bit* do vetor de probabilidades será atualizado apenas se os alelos de cada indivíduo, referentes a tal *bit*, forem diferentes. Neste caso, se um indivíduo é melhor nos dois objetivos, então o vetor de probabilidades será atualizado duas vezes com um passo de tamanho $\frac{1}{n}$ no sentido do indivíduo vencedor. Caso o indivíduo vencedor seja diferente em cada um dos objetivos, então o vetor de probabilidades permanecerá inalterado. Isso porque a atualização promovida pelo $vencedor_f_2$ será exatamente o contrário da atualização promovida pelo $vencedor_f_1$. A Figura 3.1 exemplifica o funcionamento do AGC-D, ilustrando a primeira iteração do algoritmo.

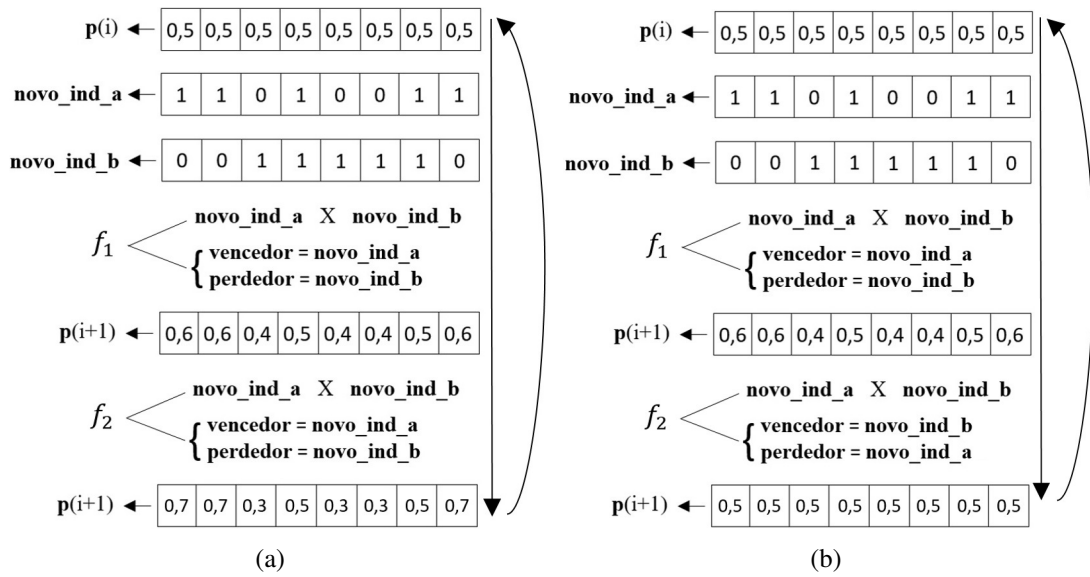


Figura 3.1: Exemplo de funcionamento do AGC-D: (a) exemplifica o caso quem que uma solução foi melhor nos dois objetivos avaliados; (b) exemplifica o caso quem que uma solução foi melhor em no primeiro objetivo e a outra solução foi melhor no segundo objetivo.

Na Figura 3.1(a), a solução **novo_ind_a** apresentou maior aptidão nas duas funções avaliadas. Ao final da iteração, o vetor de probabilidades evoluiu no sentido de aumentar a probabilidade gerar soluções parecidas com **novo_ind_a**. Na Figura 3.1(b), por outro lado, cada solução venceu uma disputa. Neste caso, o vetor de probabilidades volta ao seu estado original, uma vez que não foi possível estabelecer relação de dominância entre as soluções avaliadas. A principal característica deste algoritmo é, portanto, a aplicação do conceito de dominância de Pareto. O vetor de probabilidade será atualizado somente se a solução vencedora dominar em todas os objetivos a solução perdedora.

3.2 Algoritmo Genético Compacto com Ponderação

Como exposto na Seção 2.5, a principal finalidade de um algoritmo multiobjetivo é encontrar um conjunto de soluções com boa diversidade e o mais próximo possível da fronteira de Pareto [13]. Contudo, em problemas reais, nem sempre existe a necessidade de prover um conjunto de soluções. Uma única solução pode ser o suficiente para solucionar o problema proposto. Porém, é desejável que exista a possibilidade de estabelecer preferências que possam direcionar a busca para determinada região da fronteira de Pareto [7]. Por isso, no sentido de permitir o estabelecimento de prioridades entre os objetivos avaliados, foi implementada a versão ponderada do AGC, denominada Algoritmo Genético Compacto com Ponderação (AGC-P). O Algoritmo 3.2 apresenta o AGC-P.

Algoritmo 3.2: Algoritmo Genético Compacto com Ponderação - AGC-P

1. Seja l o número de variáveis disponíveis.
 2. $t \leftarrow 0$;
 3. **Enquanto** ($t < l$)
 4. $p(t) \leftarrow 0,5$; //Inicialização do vetor de probabilidades $t \leftarrow t + 1$;
 5. **Fim enquanto**
 6. $t \leftarrow 0$;
 7. $w_1 \leftarrow peso_1$;
 8. $w_2 \leftarrow peso_2$;
 9. **Enquanto** ($t < \text{número máximo de gerações ou } p \text{ não tenha convergido}$)
 10. **novo_ind_a** $\leftarrow gera_indivíduo(p)$; //gera indivíduo a partir de p
 11. **novo_ind_b** $\leftarrow gera_indivíduo(p)$; //gera indivíduo a partir de p
 12. [**vencedor_f1**, **perdedor_f1** $\leftarrow compete_f1(\text{novo_ind_a}, \text{novo_ind_b})$]
 13. **p** $\leftarrow atualiza_p(w_1, \text{vencedor_f1}, \text{perdedor_f1})$ //Atualiza o vetor de probabilidades na direção do vencedor.
 14. [**vencedor_f2**, **perdedor_f2** $\leftarrow compete_f2(\text{novo_ind_a}, \text{novo_ind_b})$]
 15. **p** $\leftarrow atualiza_p(w_2, \text{vencedor_f2}, \text{perdedor_f2})$ //Atualiza o vetor de probabilidades na direção do vencedor.
 16. **Fim Enquanto**
-

O funcionamento do Algoritmo Genético Compacto com Ponderação é bastante semelhante ao do AGC-D. A principal diferença está na maneira como o vetor de probabilidades (**p**) é atualizado. Ao contrário do que ocorre no AGC-D, a etapa de atualização utilizada no AGC-P não obedece o conceito de solução Pareto ótima. Neste caso, mesmo que as soluções vencedoras sejam diferentes, a função objetivo que tiver o maior peso terá prioridade na atualização do vetor (**p**). Por exemplo, considere que os pesos sejam estabelecidos como apresentado na Igualdade (3-2) e que os indivíduos com melhor desempenho nas funções objetivo sejam diferentes.

$$w_1 = 0,4 \quad w_2 = 0,6 \quad (3-2)$$

Então, de acordo com a Relação (3-3), o vetor de probabilidade será atualizado com um passo de tamanho $\frac{0,2}{n}$ no sentido do indivíduo com melhor desempenho no segundo objetivo. Isso porque o \mathbf{p} será atualizado com um passo igual a 0,4 em uma direção seguido de outra atualização com passo igual a 0,6 na direção contrária. Dessa maneira, a ponderação direciona a busca priorizando a otimização da segunda função objetivo.

$$\mathbf{p}(i) = \begin{cases} \mathbf{p}(i) + w/n, & \text{se } \text{vencedor}(i) = 1 \text{ e } \text{vencedor}(i) \neq \text{perdedor}(i), \\ \mathbf{p}(i) - w/n, & \text{se } \text{vencedor}(i) = 0 \text{ e } \text{vencedor}(i) \neq \text{perdedor}(i), \\ \mathbf{p}(i), & \text{se } \text{vencedor}(i) = \text{perdedor}(i), \end{cases} \quad (3-3)$$

onde n é o tamanho da população simulada, $0 \leq i < l$, w é o peso estabelecido para o objetivo avaliado.

3.3 Algoritmo Genético Compacto com Soma Ponderada

Com base no método de Soma Ponderada, foi implementada uma versão do Algoritmo Genético Compacto para solucionar problemas de otimização com mais de um objetivo. O resultado de tal implementação é chamado de Algoritmo Genético Compacto com Soma Ponderada (AGC-SP) e apresentado no Algoritmo 3.3

Algoritmo 3.3: Algoritmo Genético Compacto com Soma Ponderada - AGC-SP

1. Seja l o número de variáveis disponíveis.
 2. $t \leftarrow 0$;
 3. $w_1 \leftarrow \text{peso}_1$;
 4. $w_2 \leftarrow \text{peso}_2$;
 5. **Enquanto** ($t < l$)
 6. $p(t) \leftarrow 0,5$; //Inicialização do vetor de probabilidades $t \leftarrow t + 1$;
 7. **Fim enquanto**
 8. $t \leftarrow 0$;
 9. **Enquanto** ($t < \text{número máximo de gerações ou } p \text{ não tenha convergido}$)
 10. **novo_ind_a** $\leftarrow \text{gera_individo}(p)$; //gera indivíduo a partir de p
 11. **novo_ind_b** $\leftarrow \text{gera_individo}(p)$; //gera indivíduo a partir de p
 12. $[\text{vencedor_f}_{esc}, \text{perdedor_f}_{esc}] \leftarrow \text{compete_f}_{esc}(\text{novo_ind_a}, \text{novo_ind_b})$
 13. $\mathbf{p} \leftarrow \text{atualiza_p}(\text{vencedor_f}_{esc}, \text{perdedor_f}_{esc})$ //Atualiza o vetor de probabilidades na direção do vencedor.
 14. **Fim Enquanto**
-

O funcionamento do AGC-SP é exatamente igual ao do AGC. A diferença está na formulação da função objetivo. Enquanto o AGC busca a otimização de um único

objetivo, o AGC-SP possibilita a otimização simultânea de vários objetivos agregando-os em uma única função. Outra característica do AGC-SP é a possibilidade de estabelecer prioridades entre os objetivos através do ajuste dos coeficientes de ponderação. O ajuste de tais coeficientes geralmente depende do problema considerado. Assim, a combinação inicial dos pesos precisa ser ajustada de modo a entregar soluções satisfatórias.

Material e Métodos

Neste capítulo é feita a descrição do estudo de caso em que são aplicadas as implementações propostas no Capítulo 3. São apresentadas ainda as funções objetivo, os dados do problema e a plataforma computacional utilizada.

4.1 Estudo de caso

Um dos objetivos da química analítica é a determinação quantitativa de propriedades de interesse. Para isso uma das técnicas mais utilizadas é a de calibração multivariada [33, 38].

A calibração multivariada é o processo de construção de um modelo matemático para relacionar múltiplas respostas instrumentais a propriedades de interesse de uma determinada amostra [5]. No contexto deste trabalho, as respostas instrumentais (variáveis independentes) são representadas por uma matriz \mathbf{X} com dimensão $(m \times l)$, onde m representa o número de amostras e l o número de variáveis obtidas a partir da análise instrumental. A propriedade de interesse (variável dependente), por sua vez, é representada por um vetor \mathbf{y} de dimensão $(m \times 1)$. Por exemplo, a quantidade de energia absorvida por grãos de trigo, medida por um espectrofotômetro, corresponde a uma matriz \mathbf{X} composta por m amostras e k comprimentos de onda. Tal matriz pode ser relacionada ao vetor \mathbf{y} , composto pelas concentrações de proteína presente em m grãos. Os valores presentes em \mathbf{y} são obtidos em laboratório e servem como referência para a construção do modelo de calibração, representado pela Equação (4-1).

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \quad (4-1)$$

onde $\boldsymbol{\beta} = \beta_1, \beta_2, \dots, \beta_k$ é o vetor de coeficientes a ser determinado e $\boldsymbol{\varepsilon}$ é o erro não explicado pelo modelo.

Uma técnica frequentemente utilizada na determinação dos coeficientes do modelo (4-1) é a Regressão Linear Múltipla (RLM). Dada a matriz de variáveis independen-

tes \mathbf{X} e o vetor de variáveis dependentes \mathbf{y} , o vetor de coeficientes de regressão (β) pode ser estimado através do método de mínimos quadrados, conforme a Equação (4-2).

$$\beta = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (4-2)$$

Determinados os coeficientes de regressão, é possível utilizar o modelo para prever a concentração da propriedade de interesse a partir de novas respostas instrumentais. A predição é realizada conforme a Equação (4-3).

$$\hat{\mathbf{y}} = \mathbf{X}\beta \quad (4-3)$$

Vale relembrar a existência de um erro aleatório entre a concentração predita e a concentração real da propriedade de interesse. Tal erro é dado pela Equação (4-4).

$$\varepsilon = \mathbf{y} - \hat{\mathbf{y}} \quad (4-4)$$

Apesar do seu potencial, a RLM apresenta alguns problemas que limitam sua aplicação. Um deles é que o número de amostras deve ser igual ou maior que o número de variáveis obtidas por métodos instrumentais. Contudo, com os avanços dos equipamentos utilizados nos métodos instrumentais, essa restrição dificilmente é atendida. O não atendimento de tal condição resulta em uma matriz \mathbf{X} mal condicionada, o que pode interferir diretamente na qualidade do modelo de calibração, ou seja, um modelo com baixa capacidade preditiva [24]. Outro problema relacionado à RLM é a sua propensão ao *overfitting*[2]. Tal fenômeno ocorre quando os modelos se ajustam bem aos dados de treinamento mas com alta deterioração do resultados em dados de teste [23]. Problemas dessa natureza impõem a necessidade de selecionar um subconjunto de dados que permita a construção de modelos matemáticos mais robustos e com melhor capacidade de predição.

4.2 Conjunto de Dados

Neste trabalho foram utilizados dados provenientes de amostras de grãos de trigo obtidos a partir de material vegetal produzido por produtores canadenses. Os dados de referência foram determinados no Laboratório de Pesquisa de Grãos (*Grain Research Laboratory*), em Winnipeg. Tal conjunto de dados foi utilizado como desafio na Conferência Internacional de Reflectância Difusa, em 2008 [15].

Assim como em Soares *et al.* [46] e Lucena *et al.* [11], a concentração de proteína nas amostras de trigo foi escolhida como a propriedade de interesse. Os espectros foram adquiridos por espectrofotômetro com resolução entre 400 e 2500 nm (nanômetros) com

resolução de 2 nm. No presente trabalho, apenas a faixa entre 1100 e 2500 nm do NIR (*Near Infrared*) foi utilizada.

No pré-processamento dos dados, foi calculada a primeira derivada dos espectros utilizando o filtro de Savitzky-Golay com um polinômio de segunda ordem e janela de 11 pontos. O espectro derivativo resultante, composto por 690 variáveis espectrais, é apresentado na Figura 4.1.

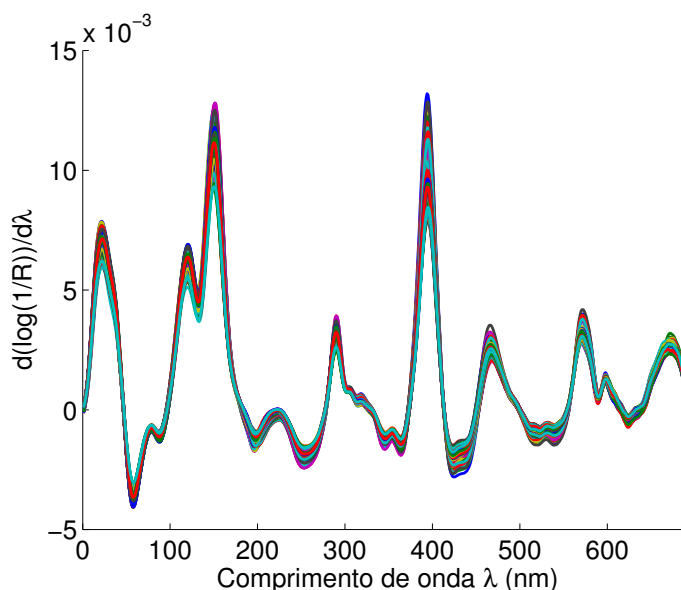


Figura 4.1: Espectro derivativo das 775 amostras e trigo.

O algoritmo Kennard-Stone (KS) foi aplicado ao espectro resultante, dividindo os dados em conjunto de calibração, validação e predição, contendo 389, 193 e 193 amostras, respectivamente. Os conjuntos de calibração e validação foram utilizados no processo de seleção de variáveis e determinação dos melhores indivíduos. O conjunto de predição foi utilizado apenas na avaliação do desempenho dos algoritmos.

4.3 Forma de Análise dos Resultados

Na aplicação do AGC e do AGC-D, os únicos parâmetros a serem definidos previamente são o tamanho da população (n) e a quantidade máxima de gerações. A definição de tais parâmetros ocorreu conforme descrito na Seção 5.1.

Considerando a aplicação dos algoritmos AGC-P e AGC-SP, além dos parâmetros já mencionados, é preciso ainda definir os pesos a serem aplicados às funções de modo a direcionar o processo de busca. Neste caso, a estratégia utilizada foi igualar a soma dos

pesos a 2, conforme apresentado na Equação 4-5.

$$\sum_{i=1}^l w_i = 1 \quad (4-5)$$

Utilizar a Equação (4-5) ao invés da Equação (2-10) justifica-se pelo processo de convergência do vetor de probabilidades (\mathbf{p}). Como definido em Harik *et al.* [26], o o passo de atualização de \mathbf{p} deve ser igual a $1/l$. Caso a Equação (2-10) fosse adotada, um dos passos de atualização seria sempre menor ou igual a $0,5/l$, o que impediria a convergência de \mathbf{p} em uma solução explícita antes de alcançar o número máximo de gerações.

A avaliação dos modelos determinados pela Regressão Linear Múltipla foram realizadas utilizando dois parâmetros. O primeiro relacionado a capacidade de predição do modelo e o outro a quantidade de variáveis selecionadas. A capacidade de predição do modelo foi aferida com base na raiz quadrada do erro quadrático médio de predição (*Root Mean Square Error of Prediction* - RMSEP), descrita pela Equação (4-6).

$$RMSEP = \sqrt{\frac{\sum_{i=1}^m (\hat{y}_i - y_i)^2}{m}} \quad (4-6)$$

, onde y_i é a concentração real da propriedade de interesse presente na amostra e \hat{y}_i é a concentração predita.

A quantidade de variáveis selecionadas é contabilizada como o total de variáveis indicada pelo vetor de probabilidades \mathbf{p} , conforme a Equação (4-7).

$$QV = \sum_{i=1}^k p_i \quad (4-7)$$

4.4 Plataforma Computacional

Todos os testes foram realizados em um computador equipado com processador *Intel core i7 3537-U* (2,5 GHz), 8 GB de memória RAM e *Windows® 10 Professional*. O software *Matlab® R2012a* (7.14.0.739) foi utilizado tanto na seleção das variáveis quanto na avaliação dos algoritmos propostos.

Resultados

Neste capítulo são apresentados e discutidos os resultados da avaliação dos algoritmos propostos. Inicialmente é explicado como foi determinado o tamanho da população simulada. Em seguida os algoritmos propostos são avaliados entre si e com outros presentes na literatura. Por fim, os algoritmos são avaliados em relação a robustez.

5.1 Determinação do tamanho da população

O tamanho da população é um dos principais parâmetros que influenciam os algoritmos evolutivos. Uma população muito pequena pode provocar, por exemplo, convergência prematura. Por outro lado, uma população muito grande aumenta o tempo de convergência do algoritmo [31, 39, 41]. Assim, escolher o tamanho adequado para a população é crucial para obter o melhor desempenho dos algoritmos propostos. Nesse sentido, o AGC-D foi executado com populações entre $n = 75$ e $n = 300$, variando em intervalos de 7,5. A Figura 5.1 mostra os resultados relacionados à quantidade de variáveis selecionadas enquanto a Figura 5.2 apresenta os resultados da minimização do erro de predição. Os valores apresentados são referentes à média de 30 simulações.

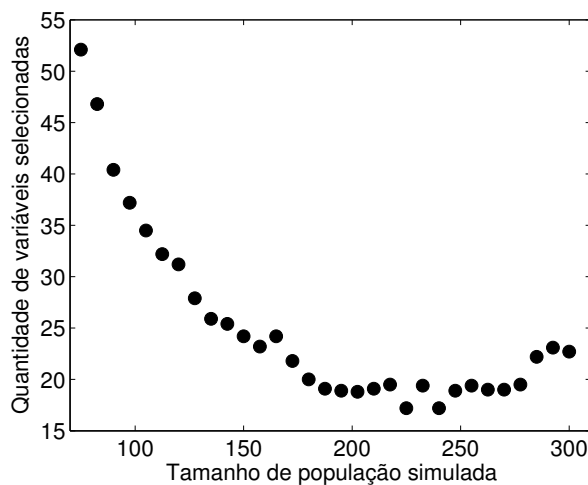


Figura 5.1: *Quantidade de variáveis selecionadas para diferentes tamanhos de população simulada.*

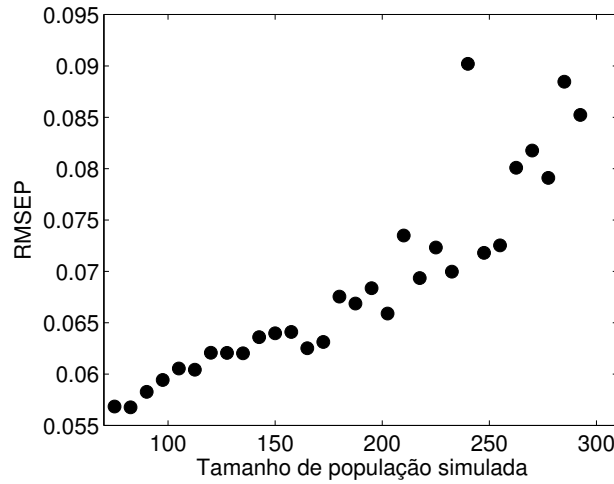


Figura 5.2: Erro de predição para diferentes tamanho de população simulada.

Como pode ser observado nas Figuras 5.1 e 5.2, o tamanho da população influencia diretamente a otimização dos objetivos escolhidos. Populações maiores priorizam a redução da quantidade de variáveis selecionadas, enquanto populações menores favorecem a minimização do RMSEP. Uma observação adicional quanto à Figura 5.1 é que a partir da população de tamanho 200, verifica-se o aumento da quantidade de variáveis selecionadas, enquanto o esperado era a diminuição desse valor. Esse fato ocorre pois, para populações com tamanho maior que 200, o algoritmo é finalizado antes da convergência do vetor de probabilidades por conta da limitação do número de gerações.

No sentido de estabelecer maior parcimônia entre os objetivos avaliados, a Equação 5-1 foi utilizada para selecionar o tamanho de população mais adequado. Dada a diferença de escala entre os valores apresentados nas Figuras 5.1 e 5.2, fez-se necessárias a normalização de tais valores antes do cálculo da parcimônia.

$$parcimonia = \frac{1}{n_e} \sum_{i=1}^{n_e} (|RMSEP_{norm(i)} - QV_{norm(i)}|), \quad (5-1)$$

onde n_e equivale à quantidade de simulações realizadas.

A Equação 5-1 foi utilizada para encontrar a menor diferença possível entre os objetivo avaliados. Quanto mais próximo de zero, maior o equilíbrio entre os objetivos. A Figura 5.3 mostra o resultado obtido em cada simulação, considerados os diferentes tamanhos de população avaliados.

Como apresentado na Figura 5.3, quando a população simulada é igual a 120 a prioridade entre os objetivos é praticamente igual. Portanto, esse foi o tamanho da população escolhido para simular e comparar a eficiência dos algoritmos propostos neste trabalho.

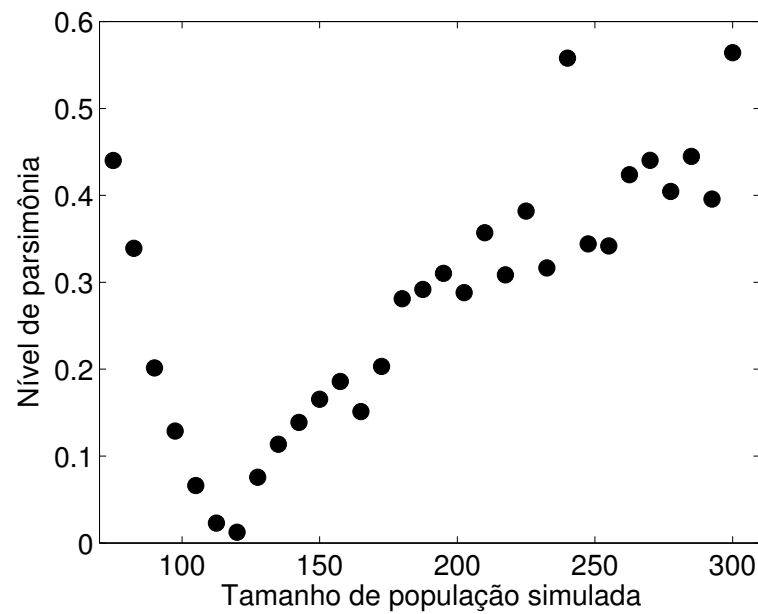


Figura 5.3: *Nível de parcimônia entre a quantidade de variáveis selecionadas e o erro de predição para diferentes tamanhos de população.*

5.2 Resultados para o AGC-D

Definido o tamanho da população, o AGC-D e o AGC foram executados com os seguintes parâmetros:

- População simulada (n) = 120 indivíduos
- Número máximo de gerações = 10000

A implementação do AGC tem por objetivo apenas a minimização do RMSEP enquanto o AGC-D utiliza também o objetivo de minimizar a quantidade de variáveis selecionadas. A Tabela 5.1 apresenta os resultados obtidos pelos dois algoritmos e os resultados do AGC com operador de mutação (*Mutation-based Compact Genetic Algorithm - mCGA*) apresentados em Soares *et al.*[46].

	AGC		AGC-D		mCGA	
Função objetivo	QV	RMSEP	QV	RMSEP	QV	RMSEP
Média	164,9	0,056	30,97	0,059	80	0,10
Mínimo	139	0,043	23	0,049	40	0,06
Máximo	192	0,063	39	0,072	120	0,18

Tabela 5.1: *Desempenho dos algoritmos em relação à quantidade de variáveis selecionadas e ao erro de predição.*

Considerando o mCGA e a quantidade de variáveis selecionadas, o AGC-D teve um desempenho aproximadamente 61,2% melhor. Já em relação ao RMSEP, o algoritmo proposto apresentou-se 40% superior. Mesmo no pior caso, quando comparado ao desempenho médio do mCGA, o AGC-D reduziu o erro de predição em 27,6%. O desempenho do algoritmo proposto pode ser justificada pela inserção de um objetivo que busca minimizar a quantidade de variáveis selecionadas. Neste caso, o algoritmo é forçado a selecionar variáveis mais significativas para minimizar os dois objetivos simultaneamente.

Comparado ao AGC, o AGC-D proporcionou uma redução de 81,3% no número de variáveis selecionadas. Por outro lado, o resultado em relação ao RMSEP foi ligeiramente pior, cerca de 5%. Neste caso, a superioridade do AGC pode ser justificada pelo elevado número de variáveis utilizadas no modelo de calibração. Ao buscar exclusivamente a minimização do erro de predição, o AGC utiliza todas as variáveis que possibilitem a redução do RMSEP, ainda que tal redução seja mínima. Contudo, como discutido anteriormente, esta característica pode provocar o sobreajuste do modelo de calibração, ou seja, o *overfitting*. A hipótese de *overfitting* é analisada e discutida na Seção 5.4.

As Figuras 5.4 e 5.5 mostram as variáveis selecionadas pelos algoritmos AGC e AGC-D, respectivamente.

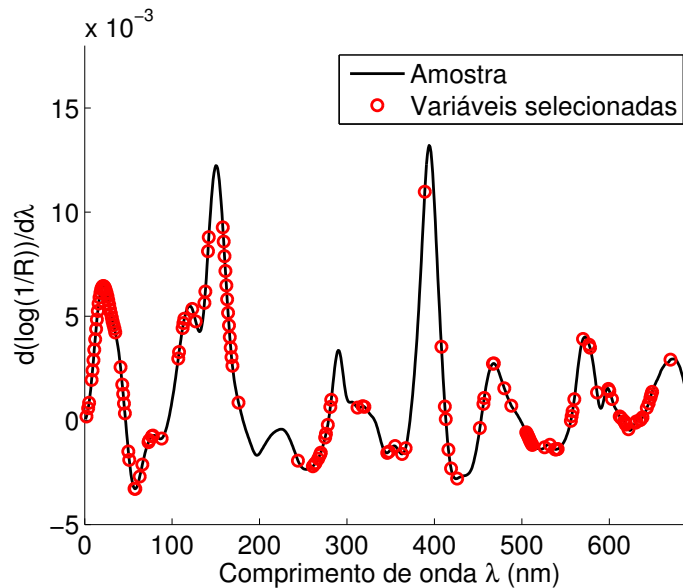


Figura 5.4: Variáveis selecionadas pelo AGC.

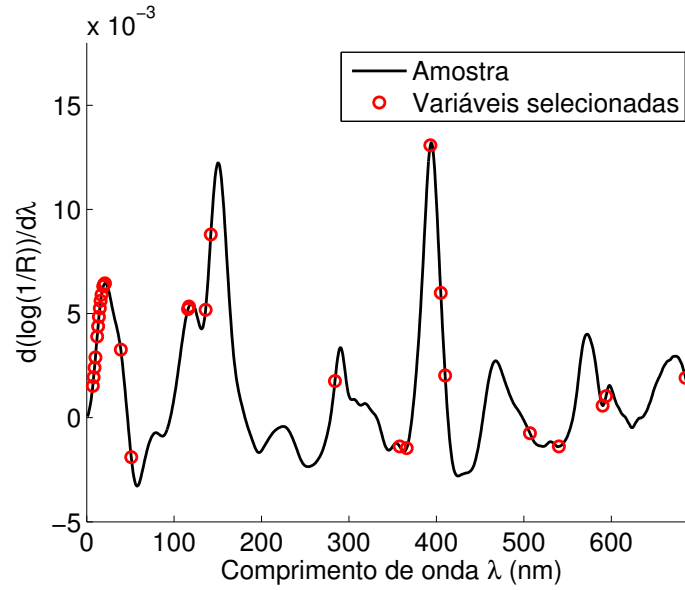


Figura 5.5: Variáveis selecionadas pelo AGC-D.

Observando as Figuras 5.4 e 5.5 é possível verificar que as variáveis selecionadas pelo AGC estão, em sua maioria, na mesma região espectral que as variáveis selecionadas pelo AGC-D. Isso indica que os dois algoritmos consideram como mais promissoras as mesmas regiões espectrais. Contudo, o AGC tende a selecionar um elevado número de variáveis, mesmo que o ganho do RMSEP não seja relevante.

5.3 Resultados para as versões ponderadas do AGC

Os Algoritmos AGC-P e AGC-SP foram simulados utilizando 9 diferentes ajustes dos coeficientes de ponderação. O coeficiente w_1 pondera o objetivo de minimizar a quantidade de variáveis selecionadas, enquanto w_2 pondera o objetivo de minimizar o erro de predição do modelo de calibração. Ambos os algoritmos foram executados 30 vezes e as médias dos resultados obtidos são apresentados na Tabela 5.2.

Os resultados apresentados na Tabela 5.2 mostram que o AGC-SP possui desempenho superior ao do AGC-P na minimização dos dois objetivos avaliados. Em relação à quantidade de variáveis selecionadas, o AGC-SP foi 17,43% superior ao AGC-P na média de todas as combinações de pesos. Considerando o RMSEP, o desempenho do AGC-SP foi aproximadamente 20% melhor em média. Portanto, entre as duas versões ponderadas do AGC, o AGC-SP apresenta-se como uma alternativa relativamente melhor. Uma possível justificativa para o pior desempenho do AGC-P é que o vetor de probabilidades é sempre atualizado, ainda que o indivíduo vencedor seja bom em apenas um dos objetivos avaliados. Esta característica pode direcionar a busca para regiões pouco promissoras. Por fim, é possível concluir a partir da análise da Tabela 5.2 que a ponderação dos objetivos cumpre sua finalidade em ambos os algoritmos. Quando o peso é maior para um objetivo,

Coeficientes de ponderação	AGC-P		AGC-SP	
	QV	RMSEP	QV	RMSEP
$w_1 = 0,8$ e $w_2 = 1,2$	73,3	0,0564	56,13	0,0515
$w_1 = 0,85$ e $w_2 = 1,15$	66,5	0,0552	50,83	0,0497
$w_1 = 0,9$ e $w_2 = 1,1$	52,2	0,0565	42,63	0,0503
$w_1 = 0,95$ e $w_2 = 1,05$	44,4	0,0576	33,27	0,0549
$w_1 = 1$ e $w_2 = 1$	34,1	0,0604	32,5	0,0563
$w_1 = 1,05$ e $w_2 = 0,95$	28,2	0,0665	26,53	0,0592
$w_1 = 1,1$ e $w_2 = 0,9$	22,2	0,0946	22,33	0,0665
$w_1 = 1,15$ e $w_2 = 0,85$	22,9	0,0818	18,5	0,0748
$w_1 = 1,2$ e $w_2 = 0,8$	17	0,1696	15,17	0,0893

Tabela 5.2: Desempenho dos algoritmos AGC-P e AGC-SP em relação à quantidade de variáveis selecionadas e ao erro de predição.

este tem a sua otimização priorizada em detrimento do outro. Tais observações podem ser melhor visualizadas na Figura 5.6.

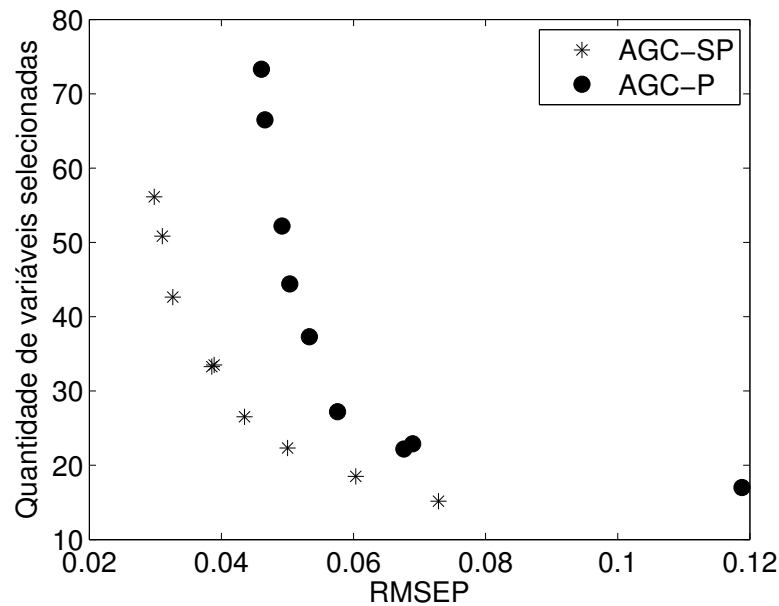


Figura 5.6: Comparação entre os algoritmos AGC-P e AGC-SP.

Visto que AGC-SP possui resultados melhores do que AGC-P, seus resultados são comparados aos obtidos por AGC-D. Para tanto, foram considerados apenas os resultados do AGC-SP quando os coeficientes de ponderação são iguais a 1, ou seja, não existe prioridade entre os objetivos. A Tabela 5.3 apresenta os resultados dos dois algoritmos.

Pode-se observar, a partir da Tabela 5.3, que o desempenho dos algoritmos, na média das 30 execuções, é similar. AGC-D em média seleciona uma quantidade menor

	AGC-D		AGC-SP	
	QV	RMSEP	QV	RMSEP
Média	30,97	0,0590	33,5	0,0563
Menor QV	21	0,0602	14	0,0785
Menor RMSEP	37	0,0498	43	0,0395
Maior QV	39	0,0567	49	0,0560
Maior RMSEP	33	0,0724	19	0,0857

Tabela 5.3: Desempenho dos algoritmos AGC-D e AGC-SP em relação à quantidade de variáveis selecionadas e ao erro de predição.

de variáveis, no entanto AGC-SP possui um RMSEP menor. Para estabelecer um melhor comparativo entre os dois algoritmos foi utilizada a Equação (5-1) para definir o nível de parcimônia de cada algoritmo. Nesta avaliação, o nível de parcimônia do AGC-SP foi de 0,2 enquanto do AGC-D foi de 0,12, ou seja, o AGC-D foi cerca de 40% superior nesse sentido .

Adicionalmente, é importante destacar que os resultados obtidos pelo AGC-SP apresentam um comportamento mais disperso que os obtidos pelo AGC-D, como pode-se observar na Figura 5.7.

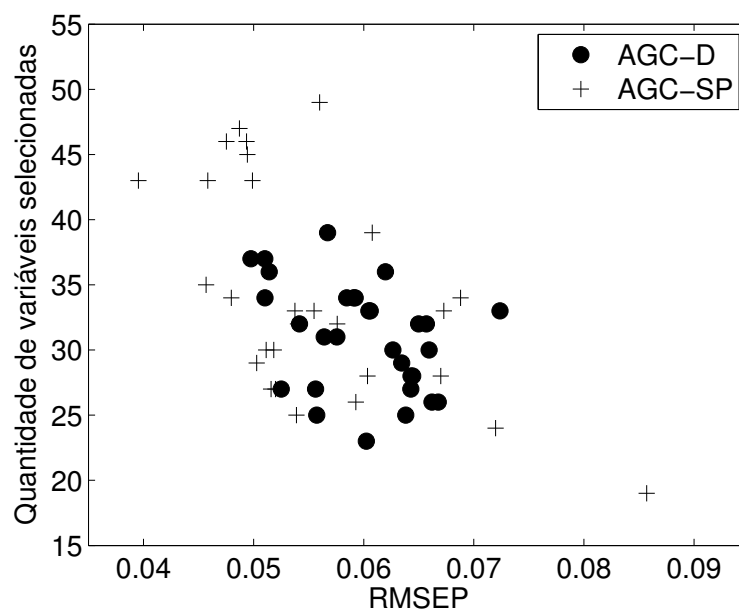


Figura 5.7: Comparação entre os algoritmos AGC-D e AGC-SP.

Em relação ao RMSEP, os resultados do AGC-SP apresentam desvio padrão de 0,0211 enquanto do AGD-D é de 0,0095. O mesmo ocorre em relação à quantidade de variáveis selecionadas. Enquanto o desvio padrão no AGC-SP é de 17,33, no AGC-D é

de 8, 1. Tais observações indicam que o conceito de dominância aplicado ao AGC tende a aumentar a confiabilidade e a precisão dos resultados obtidos.

5.4 Avaliação da Robustez dos Algoritmos Propostos

Conforme destacado na Seção 5.2, devido à grande quantidade de variáveis selecionadas, os resultados obtidos pelo AGC indicam a existência de *overfitting*. Considerando ainda a pouca precisão dos resultados obtidos pelo AGC-SP, optou-se por avaliar o desempenho dos algoritmos propostos na presença de ruídos instrumentais. Para esse experimento, um sinal de ruído da ordem de 10^{-10} foi gerado a partir da multiplicação da variância dos dados originais por valores aleatórios gerados por meio da função *randn()* do Matlab. Em seguida, o sinal ruidoso foi adicionado aos dados de teste. Para ilustrar a amplitude do ruído gerado, destaca-se que a variância dos sinais originais é da ordem de 10^{-8} .

Construído o novo conjunto de teste a partir da adição de ruídos, os Algoritmos AGC, AGC-D e AGC-SP foram submetidos ao processo de avaliação. Cada algoritmo foi executado 30 vezes. A Figura 5.8 apresenta os resultados do ACG.

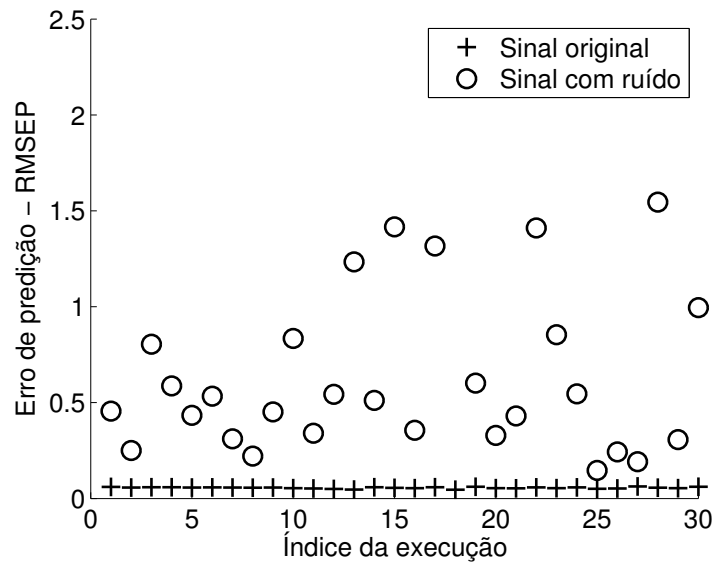


Figura 5.8: Avaliação do AGC na presença de dados ruidosos.

Ao analisar a Figura 5.8 é possível observar que os modelos obtidos a partir das variáveis selecionadas pelo AGC geram *overfitting* no modelo. Mesmo na presença de ruídos de pequena amplitude, o RMSEP do modelo de calibração determinado pelo algoritmo foi mais de 1300% pior. Na média, o RMSEP foi de 0,7604.

A Figura 5.9 apresenta os resultados da simulação do ACG-SP. Mesmo com resultados melhores que o apresentado pelo AGC, o AGC-SP também demonstrou pouca

robustez na presença de ruídos. Com RMSEP médio de 0,325, o AGC-SP apresentou desempenho 570% pior quando comparado aos valores obtidos com os dados originais.

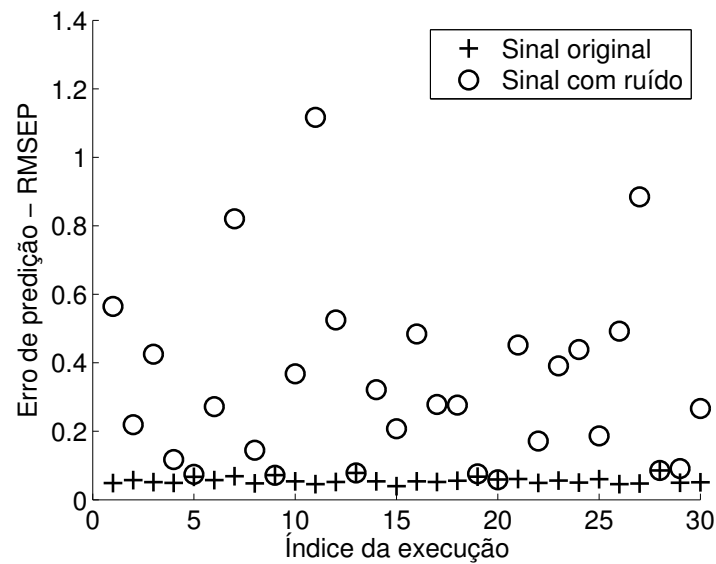


Figura 5.9: Avaliação do AGC-SP na presença de dados ruidosos.

Por fim, a Figura 5.10 apresenta os resultados obtidos a partir da simulação do ACG-D.

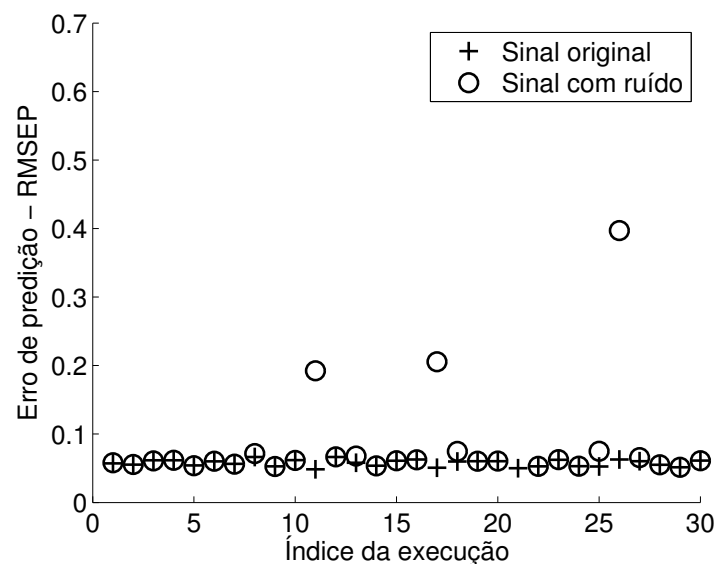


Figura 5.10: Avaliação do ACG-D na presença de dados ruidosos.

Analisando a Figura 5.10 é possível verificar que os resultados do ACG-D tiveram baixa capacidade de generalização em 3 execuções. Ou seja, 10% das simulações. Na maioria dos casos, o algoritmo mostrou-se robusto à presença de ruídos. Na média, o ACG-D obteve RMSEP igual a 0,0853. Dessa maneira, é possível destacar a robustez do

algoritmo para trabalhar com dados ruidosos, ao contrário dos algoritmos apresentados anteriormente.

Conclusões

Diversas estratégias de seleção de variáveis foram desenvolvidas e apresentadas na literatura. Apesar dos avanços, características indesejáveis como a multicolinearidade continuam a prejudicar a acurácia de classificadores e regressores. As estratégias de seleção de variáveis do tipo *wrapper*, que fazem uso de apenas um objetivo, costumam provocar o *overfitting*. Tal característica diminui a capacidade de generalização de classificadores e regressores. Nesse contexto, este trabalho propôs o Algoritmo Genético Compacto com Dominância. Tal proposta utiliza uma abordagem alternativa para representar o conjunto de soluções e conduzir a busca pelo espaço de soluções.

No estudo de caso foi avaliado a acurácia dos classificadores e regressores a partir de um subconjunto de variáveis selecionado pelo algoritmo proposto. Com base nos resultados obtidos, pode-se dizer que o AGC-D foi capaz de selecionar um subconjunto de variáveis menor e mais representativo. Quando comparado com a estratégia monoobjetivo apresentada em Soares *et al.* [46], o erro de predição do regressor foi cerca de 40% menor, enquanto o número de variáveis selecionadas foi reduzido em mais de 61,2%. Quando comparado à implementação do Algoritmo Genético Compacto com Soma Ponderada, também apresentada neste trabalho, o desempenho do AGC-D foi ligeiramente inferior no quesito erro de predição e equivalente quanto ao número de variáveis selecionadas. Contudo, o AGC-D apresentou maior parcimônia entre os objetivos avaliados, além de demonstrar maior robustez ao ser testado com um conjunto de amostras contaminado por ruído normal. Portanto, concluiu-se que o AGC-D, proposto neste trabalho, apresenta características desejáveis de uma estratégia de seleção de variáveis, equiparando-se a outras técnicas em nível de acurácia e adicionando robustez ao regressor.

6.1 Trabalhos Futuros

Como trabalho futuro, propõe-se avaliar o Algoritmo Genético Compacto com Dominância em outros conjuntos de dados e com outros classificadores e regressores. Adicionalmente, entende-se que a função objetivo que quantifica a quantidade de variáveis selecionadas pode ser melhor desenvolvida. Isto porque tal função é de primeira or-

dem, enquanto a função que contabiliza o erro de predição é de segunda ordem. Assim, o AGC-D, frequentemente, tende a minimizar a quantidade de variáveis mais rapidamente, prejudicando a acurácia do regressor. Por fim, uma abordagem que solucione o problema dos *building blocks* ainda é um desafio em aberto na ciência.

Referências Bibliográficas

- [1] **A survey on feature selection methods.** *Computers and Electrical Engineering*, 40.
- [2] **The problem of overfitting.** *Journal of Chemical Information and Computer Sciences*, 44(1):1–12, 2003.
- [3] ANDERSSON, J. **A survey of multiobjective optimization in engineering design** johan, 2000.
- [4] BEASLEY, D.; BULL, D. R.; MARTIN, R. R. **An overview of genetic algorithms : Part 1, fundamentals**, 1993.
- [5] BEEBE, K. R.; PELL, R. J.; SEASHOLTZ, M. B. **Chemometrics: a practical guide.** Wiley-Interscience Series on Laboratory Automation, 1998.
- [6] BING XUE, MENGJIE ZHANG, W. N. B.; YAO, X. **A survey on evolutionary computation approaches to feature selection.** *IEEE Transactions on Evolutionary Computation*, 20(4):606–626, 2016.
- [7] COELLO, C. A. **Evolutionary Multi-Objective Optimization: A Historical View of the Field.** *Computational Intelligence Magazine, IEEE*, (February 2006):28–36, 2006.
- [8] COELLO, C. A. C.; LAMONT, G. B.; VELDHIJZEN, D. A. V. **Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation).** Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [9] DASH, M.; LIU, H. **Feature selection for classification.** *Intelligent Data Analysis*, 1:131–156, 1997.
- [10] DE ALMEIDA RIBEIRO, L.; DA SILVA SOARES, A.; DE LIMA, T. W.; JORGE, C. A. C.; DA COSTA, R. M.; SALVINI, R. L.; COELHO, C. J.; FEDERSON, F. M.; GABRIEL, P. H. R. **Multi-objective genetic algorithm for variable selection in multivariate classification problems: A case study in verification of biodiesel adulteration.** In: *Proceedings of the International Conference on Computational Science, ICCS*

- 2015, *Computational Science at the Gates of Nature, Reykjavík, Iceland, 1-3 June, 2015, 2014*, p. 346–355, 2015.
- [11] DE LUCENA, D. V.; DE LIMA, T. W.; DA SILVA SOARES, A.; DELBEM, A. C. B.; FILHO, A. R. G.; COELHO, C. J.; LAUREANO, G. T. **Multi-objective evolutionary algorithm for variable selection in calibration problems: A case study for protein concentration prediction**. In: *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20-23, 2013*, p. 1053–1059, 2013.
- [12] DE PAULA, L.; SOARES, A. S.; DE LIMA, T. W.; COELHO, C. J.; OTHERS. **Variable selection for multivariate calibration in chemometrics: A real-world application with building blocks disruption problem**. In: *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, p. 1031–1034. ACM, 2016.
- [13] DEB, K. **Multi-objective genetic algorithms: Problem difficulties and construction of test problems**. *Evolutionary Computation*, 7:205–230, 1999.
- [14] DEB, K.; KALYANMOY, D. **Multi-Objective Optimization Using Evolutionary Algorithms**. John Wiley & Sons, Inc., New York, NY, USA, 2001.
- [15] DIFUSA, C. I. D. R.
- [16] E S. N. DEEPA, S. N. S. **Introduction to Genetic Algorithms**. Springer, 2008.
- [17] FARIAS, M. S. R. **Algoritmos evolucionários aplicados ao problema do caixeiro viajante multiobjetivo**. Master's thesis, Universidade Federal de Alagoas, 2008.
- [18] FRANK, J. **A study of genetic algorithms to find approximate solutions to hard 3cnf problems**. In: *Golden West International Conference on Artificial Intelligence*, 1994.
- [19] GABRIEL, P. H. R.; DELBEM, A. C. B. **Fundamentos de Algoritmos Evolutivos**. Número 75. Notas didáticas do ICMC-USP, 2008.
- [20] GEORGE, E. I. **The variable selectin problem**. *Journal of the American Statistical Association*, 95:1304–1308, December 2000.
- [21] GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization and Machine Learning**. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.
- [22] GOLDBERG, D. E. **The design of innovation: Lessons from and for competent genetic algorithms**, volume 7. Springer Science & Business Media, 2013.

- [23] GUYON, I.; ELISSEEFF, A. **An introduction to variable and feature selection.** *J. Mach. Learn. Res.*, 3:1157–1182, 2003.
- [24] GUYON, I.; ELISSEEFF, A. **Performance of some variable selection methods when multicollinearity is present.** *Chemometrics and Intelligent Laboratory Systems*, 78:103–112, 2005.
- [25] HARIK, G. **Linkage learning via probabilistic modeling in the ecga.** *Urbana*, 51(61):801–817, 1999.
- [26] HARIK, G. R.; LOBO, F. G.; GOLDBERG, D. E. **The compact genetic algorithm.** *Evolutionary Computation, IEEE Transactions on*, 3(4):287–297, 1999.
- [27] HOLLAND, J. **Adaptation in natural and artificial systems.** 1975.
- [28] JORGE, C. A. C.; DE LIMA, T. W.; DE ALMEIDA RIBEIRO, L.; FILHO, A. R. G.; COELHO, C. J.; DA SILVA SOARES, A.; DELBEM, A. C. B. **Algoritmo evolutivo multi-objetivo de tabelas para seleção de variáveis em calibração multivariada.** In: Braga, A. d. P.; Bastos Filho, C. J. A., editors, *Anais do 11 Congresso Brasileiro de Inteligência Computacional*, p. 1–7, Porto de Galinhas, PE, 2013. SBIC.
- [29] KIRA, K.; RENDELL, L. A. **A practical approach to feature selection.** In: *Proceedings of the Ninth International Workshop on Machine Learning*, ML92, p. 249–256, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.
- [30] KOHAVI, R.; JOHN, G. H. **Wrappers for feature subset selection.** *Artif. Intell.*, 97(1-2):273–324, Dec. 1997.
- [31] KOUMOUSIS, V. K.; KATSARAS, C. P. **A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance.** *IEEE Trans. Evolutionary Computation*, 10(1):19–28, 2006.
- [32] LARRAÑAGA, P. **A Review on Estimation of Distribution Algorithms**, p. 57–100. Springer US, Boston, MA, 2002.
- [33] MÁRCIA M. C. FERREIRA, ANTUNES ALEXANDRE M., M. M. S.; VOLPE, P. **Qui-miometria i: calibração multivariada, um tutorial.** *Química Nova* 2, p. 724–731, Setembro 1999.
- [34] MELANIE, M. **An Introduction to Genetic Algorithms.** The MIT Press, 1998.
- [35] MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs (3rd Ed.).** Springer-Verlag, London, UK, UK, 1996.

- [36] MITCHELL, M.; HOLLAND, J.; FORREST, S. **Relative building-block fitness and the building block hypothesis**. *D. Whitley, Foundations of Genetic Algorithms*, 2:109–126, 2014.
- [37] MIWAKO TSUJI, M. M.; AKAMA, K. **Multi-objective evolutionary algorithm for variable selection in calibration problems: A case study for protein concentration prediction**. In: *Genetic and Evolutionary Computation - GECCO, Seattle, USA, June, 2004*, p. 246–257, 2004.
- [38] NUNES, P. G. A. **Uma nova técnica para seleção de variáveis em calibração multivariada aplicada às espectrometrias US-VIS e NIR**. PhD thesis, Universidade Federal da Paraíba, 2008.
- [39] PISZCZ, A.; SOULE, T. **Genetic programming: optimal population sizes for varying complexity problems**. In: Keijzer, M.; Cattolico, M.; Arnold, D.; Babovic, V.; Blum, C.; Bosman, P.; Butz, M. V.; Coello Coello, C.; Dasgupta, D.; Ficici, S. G.; Foster, J.; Hernandez-Aguirre, A.; Hornby, G.; Lipson, H.; McMinn, P.; Moore, J.; Raidl, G.; Rothlauf, F.; Ryan, C.; Thierens, D., editors, *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, volume 1, p. 953–954, Seattle, Washington, USA, 8-12 July 2006. ACM Press.
- [40] RANA, S.; WHITLEY, D. **Genetic algorithm behavior in the MAXSAT domain**, p. 785–794. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.
- [41] ROEVA, O.; FIDANOVA, S.; PAPRZYCKI, M. **Influence of the population size on the genetic algorithm performance in case of cultivation process modelling**. In: Ganzha, M.; Maciaszek, L. A.; Paprzycki, M., editors, *FedCSIS*, p. 371–376, 2013.
- [42] ROTHLAUF, F. **Design of modern heuristics: principles and application**. Springer Science & Business Media, 2011.
- [43] ROTHLAUF, F.; GOLDBERG, D. E. **Representations for Genetic and Evolutionary Algorithms**. Physica-Verlag, 2002.
- [44] ROTHLAUF, F.; GOLDBERG, D. E. **Representations for Genetic and Evolutionary Algorithms**. Physica-Verlag, 2002.
- [45] SIEDLECKI, W.; SKLANSKY, J. **A note on genetic algorithms for large-scale feature selection**. *Pattern recognition letters*, 10(5):335–347, 1989.
- [46] SOARES, A. S.; DE LIMA, T. W.; SOARES, F. A. A. M. N.; COELHO, C. J.; FEDERSON, F. M.; DELBEM, A. C. B.; BAALEN, J. V. **Mutation-based compact genetic**

- algorithm for spectroscopy variable selection in determining protein concentration in wheat grain.** *Electronic Letters*, 50:932–934, June 2014.
- [47] SRINIVAS, N.; DEB, K. **Muiltiobjective optimization using nondominated sorting in genetic algorithms.** *Evol. Comput.*, 2(3):221–248, Sept. 1994.
- [48] VAN KERNENADE, C. H. M. **Building block filtering and fixing.** In: *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, p. 505–510, May 1998.
- [49] W. F. PUNCH, E. D. GOODMAN, M. P. L. C.-S. P. H.; ENBODY, R. **Further research on feature selection and classification using genetic algorithms.** *International Conference on Genetic Algorithm*, p. 557–564, 1993.
- [50] WEISE, T. **Global optimization algorithms. theory and application**, 2008.
- [51] WHITLEY, D. **A genetic algorithm tutorial.** *Statistics and Computing*, 4:65–85, 1994.
- [52] YANG, J.; HONAVAR, V. **Feature selection using genetic algorithms.** *IEEE Intelligent Systems*, p. 44–49, April 1998.
- [53] YU, X.; GEN, M. **Introduction to Evolutionary Algorithms.** Springer Publishing Company, Incorporated, 2012.
- [54] ZITZLER, E.; THIELE, L. **An evolutionary algorithm for multiobjective optimization: The strength pareto approach**, 1998.