UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
BACHARELADO EM ENGENHARIA ELÉTRICA

FERNANDO CALIXTO CURI
DIEGO SOLAK CASTANHO

DESENVOLVIMENTO DE UM FRAMEWORK PARA UTILIZAÇÃO DA REGRESSÃO LINEAR MÚLTIPLA COM ALGORITMOS DE OTIMIZAÇÃO E INTERFACE GRÁFICA

TRABALHO DE CONCLUSÃO DE CURSO

PONTA GROSSA
2021

FERNANDO CALIXTO CURI

DIEGO SOLAK CASTANHO

DESENVOLVIMENTO DE UM FRAMEWORK PARA UTILIZAÇÃO DA REGRESSÃO LINEAR MÚLTIPLA COM ALGORITMOS DE OTIMIZAÇÃO E INTERFACE GRÁFICA

DEVELOPMENT OF A FRAMEWORK FOR USING MULTIPLE LINEAR REGRESSION WITH OPTIMIZATION ALGORITHMS AND GRAPHICAL INTERFACE

Trabalho de Conclusão de Curso apresentado como requisito para obtenção do título de Bacharel/Bacharel em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná (UTFPR).

Orientador: Prof. Dr. Hugo Valadares Siqueira

PONTA GROSSA

2021

Ministério da Educação
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEP. ACADEMICO DE ELETRONICA-PG

TERMO DE APROVAÇÃO

TRABALHO DE CONCLUSÃO DE CURSO - TCC

DESENVOLVIMENTO DE UM FRAMEWORK PARA UTILIZAÇÃO DO REGRESSÃO LINEAR MÚLTIPLA COM ALGORITMOS DE OTIMIZAÇÃO E INTERFACE GRÁFICA

Por

DIEGO SOLAK CASTANHO

e

FERNANDO CALIXTO CURI

Monografia apresentada às 16 horas 00 min. do dia 06 de maio de 2021 como requisito parcial, para conclusão do Curso de Engenharia Elétrica da Universidade Tecnológica Federal do Paraná, Câmpus Ponta Grossa. Os candidatos foram arguidos pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação e conferidas, bem como achadas conforme, as alterações indicadas pela Banca Examinadora, o trabalho de conclusão de curso foi considerado APROVADO.

Banca examinadora:

Profª. Drª. Fernanda Cristina Côrrea	Membro
Profª. Drª. Yara de Souza Tadano	Membro
Prof. Dr. Hugo Valadares Siqueira	Orientador
Prof. Dr. Josmar Ivanqui	Professor(a) responsável TCCII

Documento assinado eletronicamente por (Document electronically signed by) **HUGO VALADARES SIQUEIRA**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em (at) 06/05/2021, às 18:42, conforme horário oficial de Brasília (according to official Brasilia-Brazil time), com fundamento no (with legal based on) art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por (Document electronically signed by) **FERNANDA CRISTINA CORREA**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em (at) 06/05/2021, às 18:43, conforme horário oficial de Brasília (according to official Brasilia-Brazil time), com fundamento no (with legal based on) art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por (Document electronically signed by) **YARA DE SOUZA TADANO**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em (at) 07/05/2021, às 16:52, conforme horário oficial de Brasília (according to official Brasilia-Brazil time), com fundamento no (with legal based on) art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por (Document electronically signed by) **JOSMAR IVANQUI**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em (at) 10/05/2021, às 10:26, conforme horário oficial de Brasília (according to official Brasilia-Brazil time), com fundamento no (with legal based on) art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site (The authenticity of this document can be checked on the website) https://sei.utfpr.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador (informing the verification code) **2009558** e o código CRC (and the CRC code) **122A0147**.

Dedicamos este trabalho as nossas famílias e amigos.

AGRADECIMENTOS

Este trabalho não poderia ser terminado sem a ajuda de diversas pessoas e a esta instituição às quais presta-se homenagem. Certamente esses parágrafos não irão atender a todas as pessoas que fizeram parte dessa importante fase. Portanto, desde já pedimos desculpas àquelas que não estão presentes entre estas palavras, mas que elas estejam certas de que fazem parte dos nossos pensamentos e de nossa gratidão.

As nossas famílias, pelo carinho, incentivo e total apoio em todos os momentos.

Ao nosso orientador, que nos mostrou os caminhos a serem seguidos e pela confiança depositada.

Aos membros da banca, que dispuseram do seu tempo para contribuir com nosso trabalho.

Aos professores e colegas do departamento, que ajudaram de forma direta e indireta na conclusão deste trabalho.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq - (processos número 405580/2018-5 e 315298/2020-0) e a Fundação Araucária (processo número 51497) pelo apoio financeiro.

Enfim, a todos os que de alguma forma contribuíram para a realização deste trabalho, nosso muito obrigado. O presente trabalho não poderia ser finalizado sem a ajuda de diversas pessoas e/ou instituições às quais prestamos nossos agradecimentos.

“Nossas virtudes e nossos defeitos são inseparáveis, assim como a força e a matéria. Quando se separam, o homem deixa de existir.” TESLA, Nikola

RESUMO

CALIXTO CURI, Fernando; SOLAK CASTANHO, Diego. **DESENVOLVIMENTO DE UM FRAMEWORK PARA UTILIZAÇÃO DA REGRESSÃO LINEAR MÚLTIPLA COM ALGORITMOS DE OTIMIZAÇÃO E INTERFACE GRÁFICA** . 2021. 74 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) — Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2021.

Os avanços tecnológicos das últimas décadas trouxeram grandes oportunidades para a sociedade. O rápido desenvolvimento de novas tecnologias permitiu aos usuários comuns a aquisição de equipamentos com ampla capacidade de processamento computacional. Cada vez mais estes recursos são utilizados como uma maneira de realizar tarefas antes vistas como longas e de difícil execução, por meio de programas de computador que automatizam as funções antes realizadas manualmente. Com base em tais premissas, neste trabalho foi desenvolvido um *framework* que possibilita a aplicação do modelo linear da regressão múltipla (MLR), muito utilizado em diversos ramos da ciência. Tal *framework* permite o cálculo dos parâmetros livres utilizando metaheurísticas de otimização, a saber, otimização por enxame de partículas, algoritmo genético e evolução diferencial. No supracitado *framework* foi elaborada uma interface amigável ao usuário que ainda possui todas as funções necessárias para customização dos algoritmos e análise dos resultados. A programação do *software* foi feita em linguagem Python, que possui os recursos necessários tanto para a programação dos algoritmos quanto para a criação do *layout* da interface. Como exemplo de aplicação foi abordada uma base de dados real com vistas a estimar o número de internações por doenças respiratórias na cidade de São Paulo. Ao final do trabalho avalia-se a eficácia dos algoritmos e um comparativo sobre seu desempenho.

Palavras-chave: Algoritmos de otimização. Framework. Interfaces de usuário. Doenças Respiratórias. Previsão.

ABSTRACT

CALIXTO CURI, Fernando; SOLAK CASTANHO, Diego. **Development of a framework for using multiple linear regression with optimization algorithms and graphical interface** . 2021. 74 p. Undergraduate Thesis (Bachelor's Degree in Electrical Engineering) — Federal University of Technology — Paraná, Ponta Grossa, 2021.

The technological advances of the last decades have brought great opportunities for society. The rapid development of new technologies has allowed common users to acquire equipment with large computational processing capabilities. Increasingly, these resources are being used as a way to perform tasks previously seen as long and difficult to execute, by means of computer programs that automate functions previously performed manually. Based on these premises, this work developed a framework that enables the application of the multiple linear regression (MLR), very common in several branches of literature. It allows the optimization of free parameters using optimization metaheuristics, namely, particle swarm optimization, genetic algorithm and differential evolution. A user-friendly interface was developed and at the same time the software has all the necessary functions for customization of the algorithms and analysis of the results. The programming of the software was done in Python language, which has the necessary resources both for the programming of the algorithms and for the creation of the interface layout. As an example of application, a real database was used to estimate the number of hospitalizations for respiratory diseases in the city of São Paulo. At the end of the work, the effectiveness of the algorithms is evaluated and their performance is compared.

Keywords: Optimization algorithms. Framework. User interfaces. Respiratory Diseases. Forecasting.

LISTA DE ILUSTRAÇÕES

Figura 1 – Fluxograma de um Algoritmo Genético Clássico 18
Figura 2 – Possibilidade para o *Crossover* de Ponto 19
Figura 3 – Fluxograma para a Evolução Diferencial 20
Figura 4 – Fluxograma da Otimização por Enxame de Partículas 22
Figura 5 – Posicionamento Geográfico da Cidade em que foi realizado o estudo. 25
Figura 6 – Número de internações por doenças respiratórias entre 2014 e 2016 (São Paulo - SP) 26
Figura 7 – Layout do programa Qt Designer 5 30
Figura 8 – Layout do programa Qt Designer 5 31
Figura 9 – Abas principais 32
Figura 10 – Campo de carregamento de arquivo de dados 32
Figura 11 – Parâmetros globais dos algoritmos 33
Figura 12 – Base de dados da cidade de São Paulo 34
Figura 13 – Tipos de método de execução 34
Figura 14 – Aba *Global Settings* 35
Figura 15 – Aba do Algoritmo Genético 36
Figura 16 – Aba da Evolução Diferencial 39
Figura 17 – Aba do PSO 42
Figura 18 – Campo de carregamento dos dados resultantes 44
Figura 19 – Opções para plotagem de gráficos 45
Figura 20 – Tabela de Resultados 46
Figura 21 – Aba *Results* 47
Figura 22 – Curvas de Treinamento entre as variáveis Simuladas e Observadas 48
Figura 23 – Curvas de Teste entre as variáveis Simuladas e Observadas 49
Figura 24 – Evolução do Fitness para o Melhor Indivíduo 50
Figura 25 – Evolução do Custo para o Melhor Indivíduo 51
Figura 26 – Dispersão do Fitness para os *lags* 52
Figura 27 – Dispersão do Custo para 8 *lags* 53
Figura 28 – Curvas de Treinamento entre as variáveis Simuladas e Observadas 54
Figura 29 – Curvas de Teste entre as variáveis Simuladas e Observadas 55
Figura 30 – Evolução do Fitness para o Melhor Indivíduo 56
Figura 31 – Evolução do Custo para o Melhor Indivíduo MLR-GA 57
Figura 32 – Dispersão do Fitness para os *lags* 58
Figura 33 – Dispersão do Custo para 8 *lags* 59
Figura 34 – Curvas de Treinamento entre as variáveis Simuladas e Observadas 60
Figura 35 – Curvas de Teste entre as variáveis Simuladas e Observadas 61
Figura 36 – Evolução do Fitness para o Melhor Indivíduo 62
Figura 37 – Evolução do Custo para o Melhor Indivíduo 63
Figura 38 – Dispersão do Fitness para os *lags* 64
Figura 39 – Dispersão do Custo para 8 *lags* 65

LISTA DE TABELAS

Tabela 1 – Resultados obtidos para o *lag* 0 66
Tabela 2 – Resultados obtidos para o *lag* 1 66
Tabela 3 – Resultados obtidos para o *lag* 2 66
Tabela 4 – Resultados obtidos para o *lag* 3 67
Tabela 5 – Resultados obtidos para o *lag* 4 67
Tabela 6 – Resultados obtidos para o *lag* 5 68
Tabela 7 – Resultados obtidos para o *lag* 6 68
Tabela 8 – Resultados obtidos para o *lag* 7 68

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

SIGLAS

AE	Erro Absoluto, do inglês *Absolute Error*
AI	Inteligência Artificial, do inglês *Artificial Intelligence*
ARV	Variância Relativa Média, do inglês *Average Relative Variance*
DE	Evolução Diferencial, do inglês *Differential Evolution*
EA	Algoritmos Evolutivos, do inglês *Evolutionary Algorithms*
GA	Algoritmo Genético, do inglês *Genetic Algorithm*
GUI	Interface Gráfica de Usuário, do inglês *Graphical User Interface*
MAE	Erro Absoluto Médio, do inglês *Mean Absolute Error*
MAPE	Erro Percentual Absoluto Médio, do inglês *Mean Absolute Percentage Error*
MLR	Regressão Linear Múltipla , do inglês *Multiple Linear Regression*
MSE	Erro Quadrático Médio, do inglês *Mean Square Error*
PSO	Otimização por Enxame de Partículas, do inglês *Particle Swarm Optimization*
RMSE	Raiz do Erro Quadrático Médio, do inglês *Root Mean Squared Error*
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1 INTRODUÇÃO 13
1.1 OBJETIVOS 14
1.1.1 Objetivos Específicos 14
1.2 JUSTIFICATIVA 14
1.3 ORGANIZAÇÃO DO TRABALHO 15

2 REFERENCIAL TEÓRICO 16
2.1 REGRESSÃO LINEAR MÚLTIPLA (MLR) 16
2.2 MODELOS DE OTIMIZAÇÃO 16
2.2.1 Algoritmo Genético (GA) 16
2.2.2 Evolução Diferencial (DE) 20
2.2.3 Otimização por Enxame de Partículas (PSO) 21
2.3 FRAMEWORK COM INTERFACE GRÁFICA PARA REALIZAR OTIMIZAÇÃO 23

3 PROCEDIMENTOS METODOLÓGICOS 25
3.1 BASE DE DADOS 25
3.2 FUNÇÕES OBJETIVO E MÉTRICAS DE AVALIAÇÃO DE DESEMPENHO 26
3.2.1 Funções Objetivo 27
3.2.2 Métricas de avaliação de desempenho 27
3.3 MODELOS PARA ESTIMAÇÃO DE INTERNAÇÕES 29
3.4 NORMALIZAÇÃO DOS DADOS 29
3.5 DESENVOLVIMENTO DA INTERFACE GRÁFICA DO *FRAMEWORK* 31
3.5.1 Abas Principais 32
3.5.2 Aba Configurações Globais 32
3.5.3 Aba Algoritmo Genético (GA) 35
3.5.3.1 Opções 36
3.5.3.2 Tipos de Seleção 37
3.5.3.3 Tipos de *Crossover* 37
3.5.3.4 Tipos de Mutação 38
3.5.4 Aba Evolução Diferencial (DE) 39
3.5.4.1 Opções 39
3.5.4.2 Tipos de Seleção 40
3.5.4.3 Tipos de *Crossover* 40
3.5.4.4 Tipos de Mutação 41
3.5.5 Aba Otimização por Enxame de Partículas (PSO) 42
3.5.5.1 Opções 42
3.5.5.2 Tipos de velocidade 43
3.5.5.3 Tipos inerciais 43
3.5.5.4 Tipos de constantes 44
3.5.6 Aba de Resultados 44

4 RESULTADOS E DISCUSSÃO 48
4.1 REGRESSÃO LINEAR MÚLTIPLA OTIMIZADA PELA EVOLUÇÃO DIFERENCIAL (MLR-DE) 48

4.2 REGRESSÃO LINEAR MÚLTIPLA OTIMIZADA PELO ALGORITMO GENÉTICO (MLR-GA) 53
4.3 REGRESSÃO LINEAR MÚLTIPLA OTIMIZADA PELO ENXAME DE PARTÍCULAS (MLR-PSO) 59
4.4 COMPARATIVO DOS RESULTADOS 65
5 CONCLUSÕES E PERSPECTIVAS 70
5.1 TRABALHOS FUTUROS 70
REFERÊNCIAS 72
ÍNDICE REMISSIVO 74

1 INTRODUÇÃO

Para o alcance de previsões ou estimativas cada vez mais sofisticadas, é fundamental aprimorar as metodologias aplicadas. Uma alternativa consiste no uso do Regressão Linear Múltipla , do inglês *Multiple Linear Regression* (MLR), em que a análise da regressão consiste em determinar as correlações entre duas ou mais variáveis que tenham relações de causa e efeito (UYANIK; GÜLER, 2013).

Uma alternativa plausível para determinar os coeficientes do MLR envolve o uso de metaheurísticas bio-inspiradas, como a Otimização por Enxame de Partículas, do inglês *Particle Swarm Optimization* (PSO), Evolução Diferencial, do inglês *Differential Evolution* (DE) e Algoritmo Genético, do inglês *Genetic Algorithm* (GA), que apresentam uma capacidade de busca com potencial de atingir melhor precisão no ajuste de coeficientes (BELOTTI et al., 2020). Nestes métodos, as soluções candidatas são conhecidas genericamente como agentes.

O PSO é inspirado em teorias sócio psicológicas e no comportamento coletivo de grupos de animais. Mantém uma mesma população de agentes (partículas) durante todo o processo de otimização, possuindo uma espécie de “memória” que lhe possibilita chegar a solução desejada (CLERC; KENNEDY, 2002). Tais partículas mudam sua posição na função objetivo.

O GA busca sua inspiração na biologia dos fenômenos que decorrem na evolução das espécies, na qual o indivíduo ou cromossomo (agente) mais adaptado tende a perpetuar sua carga genética para as novas gerações, mediante a troca de seus genes (HU et al., 2009). Ideia semelhante foi utilizada na DE, a qual apresenta capacidade de minimizar funções não diferenciáveis e contínuas não lineares, inicializada com uma população uniforme e aleatoriamente gerada em que cada vetor possui genes, ou valores candidatos para cada variável.

Nos últimos tempos, o espaço para o desenvolvimento de ferramentas que facilitam a aplicações de algoritmos de otimização se torna cada vez maior, já que a demanda por *softwares* cada vez mais adaptativos e inteligentes conduzem à incorporação de novas estratégias que facilitam seu uso de forma rápida (SILVA et al., 2018).

1.1 OBJETIVOS

Este trabalho tem como objetivo geral, desenvolver um *framework* intuitivo para a estimação do número de internações por doenças epidemiológicas de forma simples, prática e rápida, mediante uma interface amigável ao usuário, apresentando resultados de forma gráfica e objetiva.

1.1.1 Objetivos Específicos

O objetivo geral será alcançado por meio dos seguintes objetivos específicos:

- Estruturar uma metodologia envolvendo MLR, otimizada pelo PSO, DE ou GA;
- Desenvolver um *framework* intuitivo para realizar previsões de doenças epidemiológicas;
- Validar o *framework* com um estudo de caso envolvendo a previsão do número de internações por doenças respiratórias na cidade de São Paulo.

1.2 JUSTIFICATIVA

A justificativa para o desenvolvimento deste estudo, está na disponibilização de uma ferramenta que auxilia na previsão de ocorrências de diversos fenômenos, utilizando e inserindo a Inteligência Artificial, do inglês *Artificial Intelligence* (AI)

Inteligência Artificial (AI) para auxiliar a comunidade em diversos setores e aspectos. São exemplos a previsão de demanda, vendas, previsão do mercado financeiro dentre outras, de forma que os dados de entradas envolvam as variáveis necessárias relacionadas ao fenômeno em questão, para que a ferramenta possa realizar a referida tarefa.

Neste estudo, a ferramenta foi utilizada em um estudo de caso para a previsão de internações por doenças epidemiológicas. Neste cenário, a ferramenta desenvolvida permite o melhor gerenciamento dos insumos hospitalares e seus recursos mediante observação dos picos de internações.

1.3 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado em seus capítulos subsequentes da seguinte forma: o Capítulo 2 contém referencial teórico deste estudo; o Capítulo 3 apresenta a metodologia que conduziu este estudo; O Capítulo 4 apresenta os resultados alcançados e, por fim, o Capítulo 5 expõe as conclusões.

2 REFERENCIAL TEÓRICO

Neste capítulo serão discutidas as bases teóricas e preceitos básicos do processo de estimação de previsões, assim como do MLR e dos algoritmos de otimização, PSO, GA E DE.

2.1 REGRESSÃO LINEAR MÚLTIPLA (MLR)

O MLR é bastante utilizado para estimar o comportamento de sistemas por meio de relações entre os valores esperados e os preditores (NEUHAUS; MCCULLOCH, 2011). Todas as formas de regressão buscam o desenvolvimento e a avaliação de um modelo matemático, entre uma variável de resposta e um conjunto de q variáveis preditoras x (LAZZARIN, 2019). Uma regressão linear múltipla é representada na forma da Equação 1.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + ... + \beta_q x_q \tag{1}$$

em que $\beta_0, \beta_1, \beta_2, \beta_3, ..., \beta q$ são os coeficientes de regressão calibrados a partir de uma base de dados.

2.2 MODELOS DE OTIMIZAÇÃO

Nesta seção são discutidas as heurísticas bio-inspiradas disponíveis no *framework* para realizar a calibração do preditores do modelo.

2.2.1 Algoritmo Genético (GA)

Para solucionar otimizações com vistas a determinação de busca por parâmetros, tem-se utilizado Algoritmos Evolutivos, do inglês *Evolutionary Algorithms* (EA), ou seja, algoritmos inspirados na moderna teoria da evolução Darwiniana, na qual uma população de agentes possui a capacidade de buscar diferentes soluções para problemas complexos (MARTINS et al., 2013). O GA é uma técnica consolidada na engenharia e computação, pois permite otimizar coeficientes de diferentes modelos,

sendo eficiente em sistemas lineares e não lineares.

A inspiração biológica tem bases nos fenômenos que decorrem na evolução das espécies, por meio de uma seleção natural dos indivíduos, no qual os mais adaptados ao meio tendem a sobreviver (HU et al., 2009; HOLLAND, 1992; MICHALEWICZ, 1996). Assim, o mais adaptados tendem a perpetuar sua carga genética com o surgimento das novas gerações, mediante a troca de seus genes. Essa ação permite propagar a carga genética de indivíduos mais fortes, ou seja, permite alcançar a melhor solução para os resultados de otimização (HU et al., 2009; HOLLAND, 1992; MICHALEWICZ, 1996).

O nível de aptidão é determinado conforme a adequação do indivíduo ao meio, de modo que quanto maior o seu valor, maior será a probabilidade de sua sobrevivência em gerações futuras (HU et al., 2009; J.H. AL GIZI et al., 2015; HASSAN et al., 2012).

Comumente, o GA é composto basicamente por três etapas, que têm como objetivo obter a solução ideal, representadas pelos genes do indivíduo melhor adaptado, sendo eles (HOLLAND, 1992):

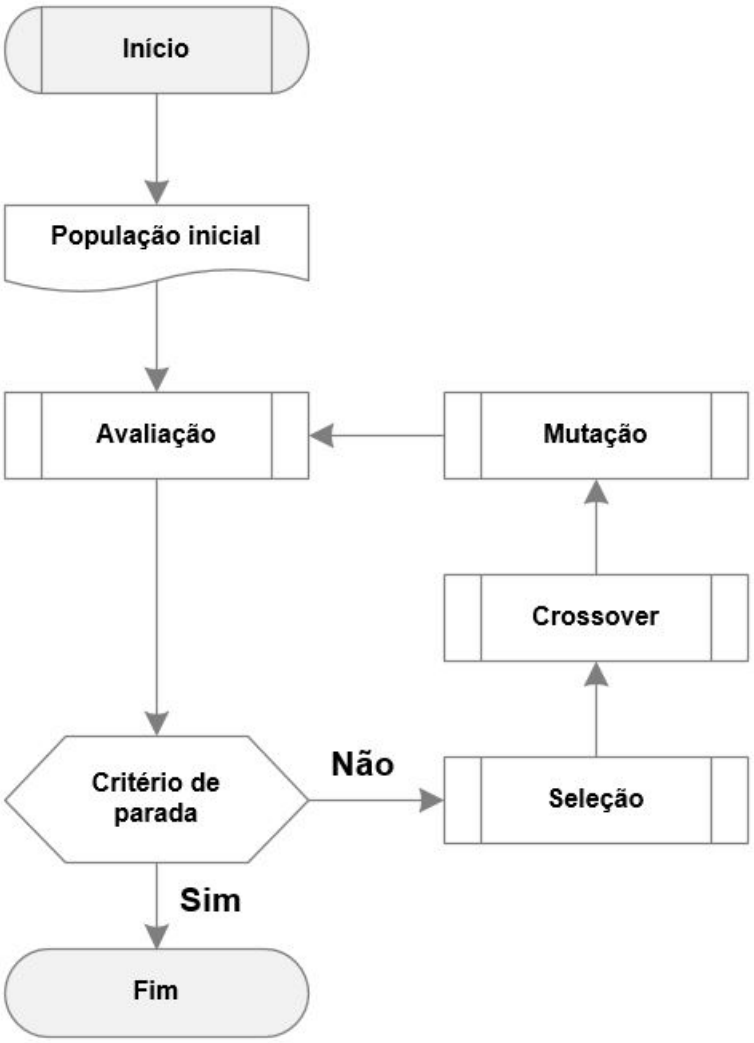
- Seleção;
- *Crossover*;
- Mutação;

Ainda, os principais parâmetros que especificam o comportamento do GA são (HU et al., 2009; ÖZTÜRK; ÇELIK, 2012; HOLLAND, 1992; MICHALEWICZ, 1996; HASSAN et al., 2012)

- Taxa de *crossover*;
- Taxa de mutação;
- Tamanho da população;
- Número de gerações.

Para melhor entendimento, a Figura 1 apresenta um fluxograma básico com a operação de um GA (CASTANHO et al., 2018).

Figura 1 – Fluxograma de um Algoritmo Genético Clássico



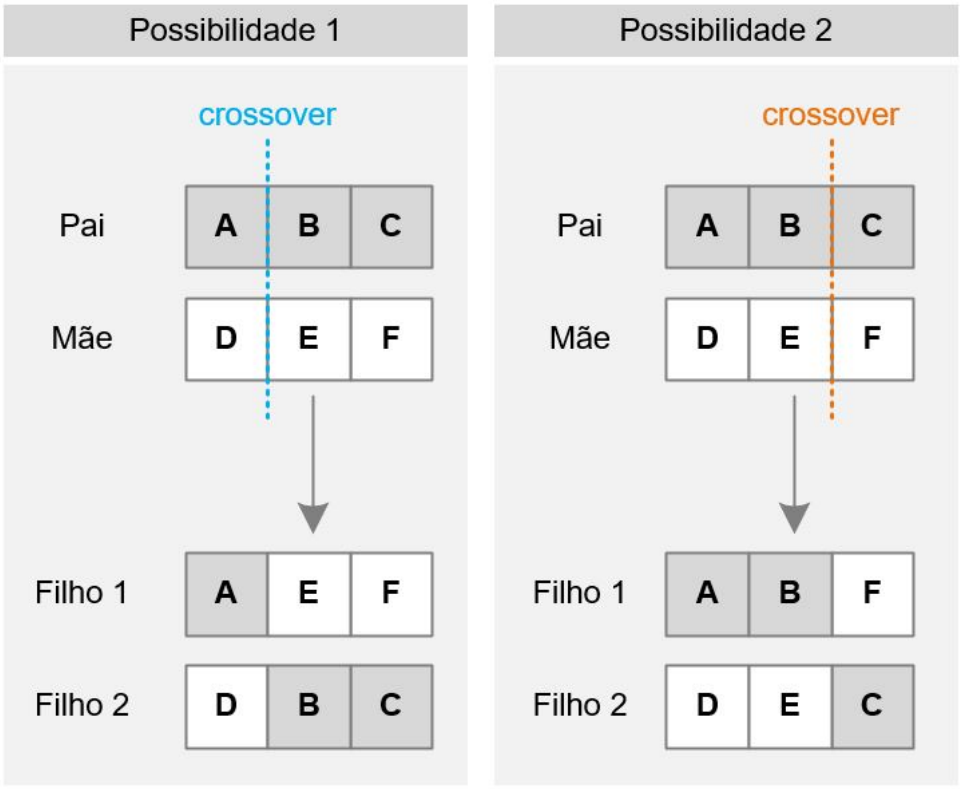
Fonte: Adaptada de Castanho et al. (2018).

As formas mais usuais para realizar a seleção, encontradas na literatura, são: (i) Seleção por Torneio; ou (ii) Seleção por Roleta. Cada abordagem apresenta diversas variações, que diferenciam-se conforme a finalidade desejada. No caso (i) forma-se um conjunto temporário selecionando aleatoriamente os cromossomos da população, em que os melhores serão direcionados à nova geração. Não analisar todos os indivíduos da população, se torna uma vantagem para este método, podendo aumentar a performance do algoritmo (HAMAMOTO, 2014). Já na roleta, a chance de cada cromossomo ser selecionado é a proporção do seu *fitness*, tomando como base o *fitness* da população inteira (HAMAMOTO, 2014).

O *crossover* é um operador genético de convergência, permitindo realizar a explotação ou busca local (DE CASTRO, 2006). O principal modelo é o *crossover* de

ponto único, em que ocorre a seleção de um ponto no cromossomo, fazendo uma troca dos genes à partir dele (HAMAMOTO, 2014). Tal ponto em que ocorre a combinação é selecionado de forma aleatória, sendo dependente apenas do número de genes (HOLLAND, 1992). A Figura 2 apresenta uma operação para dois agentes (pai e mãe) que possuem 3 genes. Observa-se na Figura 2, que há duas possibilidades para ocorrer o ponto de corte. No primeiro caso, ocorre entre A e B, enquanto no segundo, entre B e C.

Figura 2 – Possibilidade para o *Crossover* de Ponto



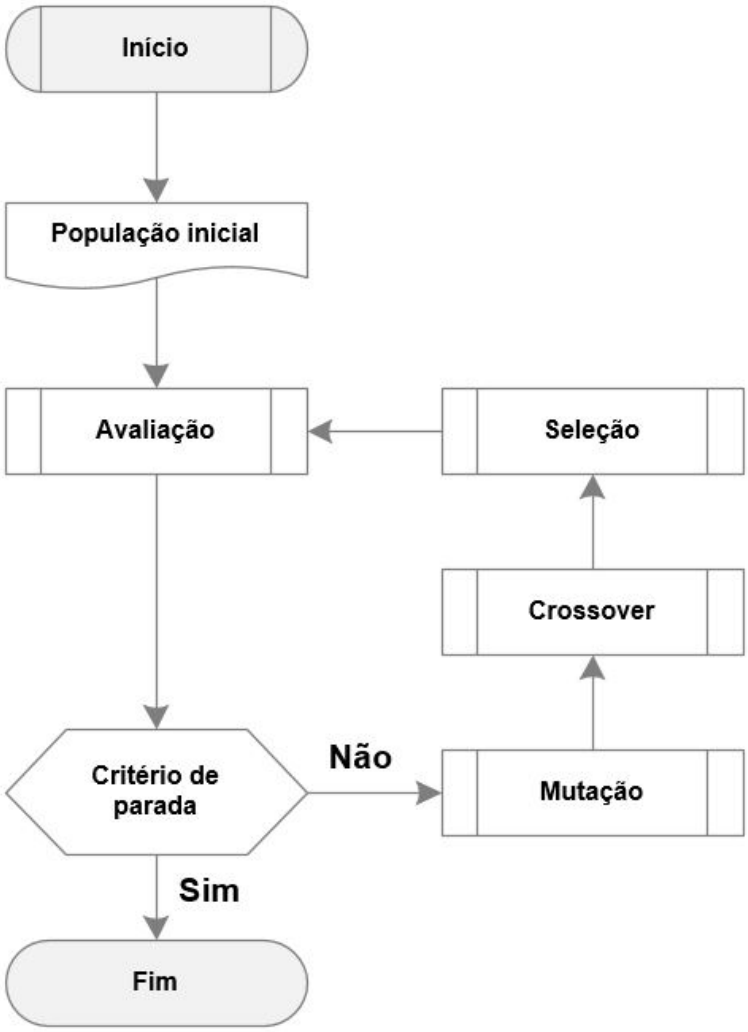
Fonte: Adaptada de Castanho (2019).

A mutação é o operador que insere no GA a capacidade de realizar uma busca global, ou seja, a exploração. Este é um operador de divergência, de caráter aleatório, pois obriga algumas soluções candidatas à explorar as regiões que ainda não foram investigadas no espaço de busca (DE CASTRO, 2006). Assim, a mutação seleciona de forma aleatória alguns genes, dentre todos os da população, gerando uma perturbação, por exemplo, Gaussiana. O parâmetro a taxa de mutação determina o percentual de genes a serem modificados ou a probabilidade de ocorrência do operador genético (HOLLAND, 1992).

2.2.2 Evolução Diferencial (DE)

A Evolução Diferencial DE, tem a capacidade de minimizar funções não diferenciáveis e contínuas não lineares, baseando-se nos operadores de Seleção, *Crossover* e Mutação (STORN; PRICE, 1997). Estes operadores são semelhantes aos encontrados no GA, porém apresentam características distintas, a começar pela ordem de apresentação, conforme apresentado na Figura 3.

Figura 3 – Fluxograma para a Evolução Diferencial



Fonte: Adaptada de Castanho et al. (2018).

Observa-se que a DE é inicializada com uma população uniforme e aleatoriamente gerada, em que cada agente (vetor) possui diferentes genes, ou valores candidatos para cada variável. Assim, de forma aleatória, um vetor é escolhido e nominado *TrialVector*.

Dentre as equações que podem ser consideradas no processo de mutação, a

Equação 2 gera o **DonorVector** para cada dimensão d:

$$DonorVector_d = MelhorIndiv_d + (F(v_{d,1} - v_{d,2})) \quad (2)$$

Na qual:

- **MelhorIndiv** representa o melhor vetor avaliado a cada iteração;
- F é uma variável que ajusta o universo de busca, sendo definida conforme a magnitude dos valores envolvidos na otimização (geralmente entre 0,1 e 2);
- $\mathbf{v}_1$ e $\mathbf{v}_2$ são vetores selecionados por meio de um processo aleatório dentro da população.

O processo de *crossover* é realizado gene a gene, resultando no *TrialVector*, com valor entre 0 e 1. Quando esse valor for menor ou igual à taxa de *crossover*, o gene do *DonorVector* é selecionado. Caso contrário, o gene do próprio *TrialVector* será escolhido.

Finalmente, o processo de seleção é aplicado entre o *TrialVector* e o indivíduo atual, por meio de uma seleção "gulosa", no qual o melhor entre os dois passará para a próxima geração. Este processo é finalizado quando a condição de parada pré estabelecida ou o número máximo de iterações sejam alcançadas (STORN; PRICE, 1997).

2.2.3 Otimização por Enxame de Partículas (PSO)

O PSO é um método é fundamentado em conceitos sociopsicológicos e na organização coletiva de grupos de animais, diferentemente dos algoritmos evolutivos, pois não se trata de um método geracional, uma vez que as partículas são mantidas durante todo o processo de otimização (KENNEDY, 2010)

O PSO realiza a busca da melhor solução mediante um ajuste da trajetória de uma partícula (agente), atualizando sua posição e velocidade, tendo por atratores a melhor posição que a partícula obteve até a iteração corrente e a da partícula melhor posicionada em sua vizinhança ou em todo o enxame (RAHMAN; ANWAR; IZADIAN, 2016)

A Figura 4 aponta as etapas gerais de execução de um PSO baseado no fluxograma apresentado no trabalho de Martins et al. (2013).

Figura 4 – Fluxograma da Otimização por Enxame de Partículas

Fonte: Adaptada de Martins et al. (2013).

Observa-se que no PSO a inicialização das partículas é realizada em posições aleatórias e com velocidades nulas. Após, realiza-se a avaliação de todas as partículas e, enquanto o critério de parada não for alcançado, atualiza-se a melhor de cada uma e a melhor posição dentre todas.

Em seguida, a atualização da velocidade de cada partícula é realizada por meio da Equação 3.

$$\mathbf{Vel} = \omega \cdot \mathbf{VelAtual} + C_{11} \cdot (\mathbf{PBest} - \mathbf{posAtual}) + C_2 \cdot r_2(\mathbf{GBest} - \mathbf{posAtual}) \quad (3)$$

Em que:

- r_1 e r_2 são números aleatórios gerados no intervalo entre 0 e 1,
- $VelAtual$ é a velocidade atual da partícula,

- $PBest$ é a melhor posição que a partícula já teve,
- $posAtual$ é a posição atual da partícula,
- $GBest$ é a melhor posição entre todas as partículas,
- ω é uma constante de ponderação de inércia,
- C_1 é a constante cognitiva e C_2 é a constante social.

Ainda, a atualização da posição da partícula é definida conforme apresentado na Equação 4.

$$\mathbf{Pos} = \mathbf{PosAtual} + \mathbf{VelAtual} \tag{4}$$

SENDO:

- **Pos** a nova posição da partícula,
- **PosAtual** a posição atual da partícula,
- **VelAtual** a velocidade atual da partícula.

2.3 FRAMEWORK COM INTERFACE GRÁFICA PARA REALIZAR OTIMIZAÇÃO

De acordo Silva et al. (2018), há uma crescente busca por ferramentas que permitam incorporam métodos mais flexíveis de otimização, bem como, que permitam alcançar melhores soluções por meio de meta-heurísticas sem exigir grandes esforços de implementação para refazer a aplicação. Os autores sugerem que pesquisadores se atenham ao desenvolvimento de estruturas como *frameworks*.

Um *framework* é capaz de fornecer uma estrutura de diferentes recursos, que visam solucionar problemas de um domínio específico, de forma rápida e fácil para o desenvolvimento de novas aplicações de otimização (SILVA et al., 2018). A literatura apresenta alguns deles para desenvolvimento de algoritmos de otimização, como o estudo de Wagner e Affenzeller (2004), que apresenta o *HeuristicLab Grid*, o qual oferece a possibilidade de uso rápido e fácil destes métodos para solucionar problemas de agendamento e planejamento de rotas.

O estudo de Wagner, Beham et al. (2010) apresenta uma nova versão do *HeuristicLab*, visando fornecer uma solução abrangente para o desenvolvimento de algoritmos, teste, análise e otimização de problemas complexos.

Por fim, Coelho et al. (2011) apresentaram o *OptFrame*, um *framework* computacional para o desenvolvimento de algoritmos eficientes, com uma interface simples

destinados à componentes de meta-heurísticas baseadas em trajetórias e populações, a fim de resolver problemas de otimização combinatória.

Neste trabalho, serão utilizadas as metodologias apresentadas neste capítulo para otimização de um modelo MLR com vistas a estimação de número de internações por doenças respiratórias.

3 PROCEDIMENTOS METODOLÓGICOS

Este capítulo apresenta os procedimentos metodológicos empregues na realização deste estudo.

3.1 BASE DE DADOS

Foi realizado um estudo de caso que envolve a estimação do número de internações hospitalares relacionadas ao contato com material particulado (MP10) para a cidade de São Paulo, localizada no estado de São Paulo, conforme a Figura 5.

Figura 5 – Posicionamento Geográfico da Cidade em que foi realizado o estudo.

Fonte: Autoria própria.

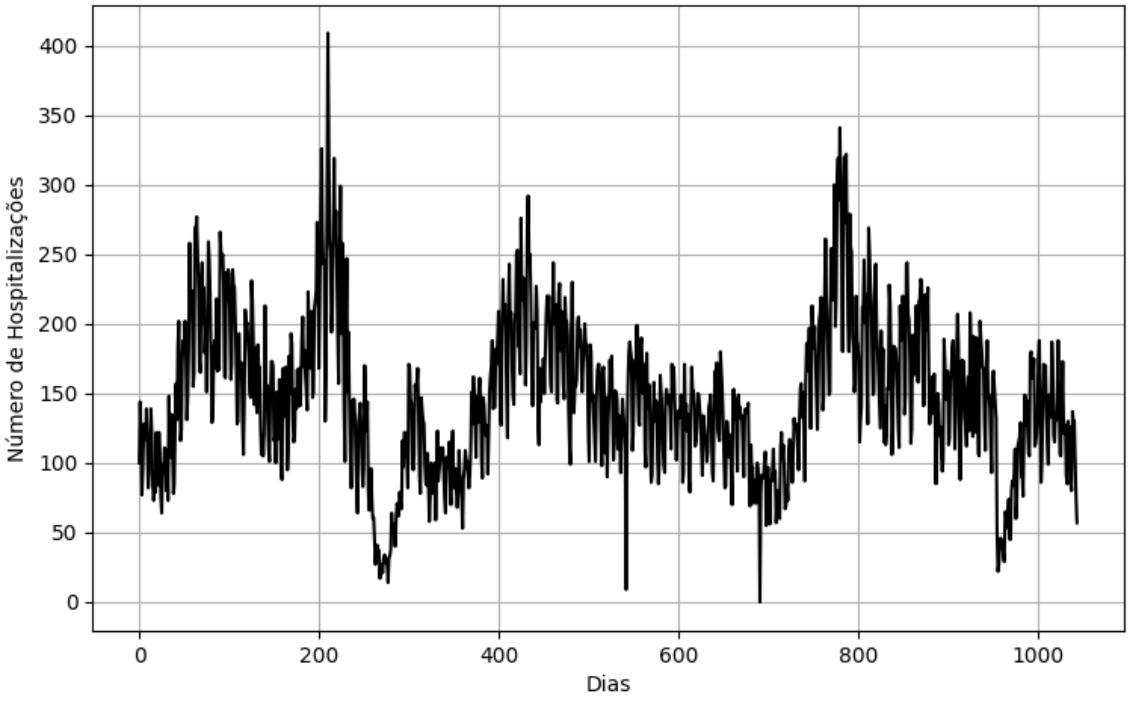
Os dados de internações por doenças respiratórias foram obtidos através da base do Sistema Único de Saúde brasileiro (DATASUS, 2016), consistindo nos dados

da Classificação Internacional de Doenças (CID-10), J00-J99.

As entradas referentes à concentração de poluição atmosférica e dados ambientais foram obtidas junto à Companhia Ambiental do Estado de São Paulo (CETESB, 2016). Como variáveis de entrada foram consideradas temperatura média, umidade relativa do ar, dia da semana e se o dia em que a internação foi registrada.

A Figura 6 apresenta as internações ocorridas no período considerado, de janeiro de 2014 até dezembro de 2016.

Figura 6 – Número de internações por doenças respiratórias entre 2014 e 2016 (São Paulo - SP)



Fonte: Autoria própria.

A base de dados de São Paulo usada para ajuste e teste dos modelos de estimação possui um total de 1045 amostras, em que foram identificados um total de 153.336 internações. Deste total, 85% das amostras foram utilizadas para treinamento e 15% para testes.

3.2 FUNÇÕES OBJETIVO E MÉTRICAS DE AVALIAÇÃO DE DESEMPENHO

Nesta seção são apresentadas as funções objetivo e métricas de avaliação de desempenho utilizadas para otimizar o modelo MLR e avaliar a qualidade dos resultados,

respectivamente.

3.2.1 Funções Objetivo

O *framework* tem disponível duas diferentes funções objetivo que podem facilmente ser configuradas pelo usuário. A primeira é o Erro Quadrático Médio, do inglês *Mean Square Error* (MSE), apresentado na Equação 5. Esta métrica é largamente difundida na literatura por ter como característica uma maior penalização para erros elevados, que, desta forma, têm maior relação ao valor final de simulação, e uma menor penalização no modelo que vier a demonstrar menores erros (MORETTIN; TOLOI, 2006):

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(d_t - y_t)^2 \tag{5}$$

em que:

- N é o número de amostras,
- d_t é o valor de internações observado,
- y_t é o valor de internações simulado.

A outra função objetivo ou função custo, é o Erro Absoluto, do inglês *Absolute Error* (AE), representado na Equação 6. Tal métrica que tem por objetivo avaliar o erro absoluto entre a curva observada e a curva matematicamente modelada através do ajuste dos coeficientes pela meta-heurística bio-inspirada.

$$AE = \sum_{t=1}^{N}|d_t - y_t| \tag{6}$$

Em que:

- N é o número de amostras,
- d_t é o valor de internações observado,
- y_t é o valor de internações simulado.

3.2.2 Métricas de avaliação de desempenho

No *framework* são calculados um total de 8 métricas de avaliação de desempenho. As duas primeiras são a MSE e o AE já apresentadas na sessão anterior. O

Erro Absoluto Médio, do inglês *Mean Absolute Error* (MAE), mostrado na Equação 7 representa o afastamento médio entre os valores reais e os valores previstos:

$$MAE = \frac{1}{N}\sum_{t=1}^{N}|d_t - y_t| \tag{7}$$

O Erro Percentual Absoluto Médio, do inglês *Mean Absolute Percentage Error* (MAPE), mostrado na Equação 8, representa uma média percentual dos erros cometidos pelo modelo em cada amostra.

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\left|\frac{d_t - y_t}{d_t}\right| \tag{8}$$

O Variância Relativa Média, do inglês *Average Relative Variance* (ARV), ilustrado na Equação 9 mostra uma medida de dispersão que determina a distância de cada amostra desse conjunto ao valor médio.

$$ARV = \sum_{t=1}^{N}\left(\frac{d_t - y_t}{y_t - \bar{d}_t}\right)^2 \tag{9}$$

em que $\bar{d}_t$ é a média dos valores observados.

Outra métrica muito usual na área ambiental e de previsão de poluentes é o Índice de concordância, do inglês *Index of Agreement* (IA), mostrado na Equação 10:

$$IA = 1 - \frac{\sum_{t=1}^{N}(d_t - y_t)^2}{\sum_{t=1}^{N}(|y_t - \bar{d}_t| + |d_t - \bar{d}_t|)^2)}, 0 \leqslant d \leq 1 \tag{10}$$

O Raiz do Erro Quadrático Médio, do inglês *Root Mean Squared Error* (RMSE), é apresentado na Equação 11. Na prática, a raiz quadrada é aplicada ao MSE, o que deixa o resultado final na mesma magnitude dos dados:

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(d_t - y_t)^2} \tag{11}$$

A média dos valores simulados é mostrada na Equação 12.

$$MEAN = \frac{d_t}{N} \tag{12}$$

Todas as métricas acima apresentarão um melhor desempenho quando minimizadas, exceto o IA que deve ser maximizado.

3.3 MODELOS PARA ESTIMAÇÃO DE INTERNAÇÕES

Neste trabalho foi considerado o modelo de regressão linear múltipla (MLR) otimizado por três diferentes técnicas bio-inspiradas: GA, DE e PSO. O *framework* desenvolvido tem implementadas duas funções objetivo como citado na Seção 3.2.1. Para esse estudo em específico utilizou-se o MSE como função custo.

Estas técnicas de otimização têm por objetivo realizar o ajuste dos coeficientes do MLR. Os modelos foram implementados na linguagem Python orientada a objetos e a interface foi desenvolvido utilizando a biblioteca PyQt5. Para que se obtivesse uma amostragem estatística de desempenho, cada algoritmo foi executado 30 vezes, de modo que a melhor execução foi escolhida.

As respectivas configurações utilizadas paras as simulações em relação aos otimizadores são:

- Algoritmo genético com população de 100 indivíduos, *crossover* de ponto a uma taxa de 70%, seleção por torneio da morte e taxa de mutação de 10%;
- Evolução diferencial no qual foram utilizados 100 vetores, *crossover* binário a uma taxa de 20%, seleção gulosa e uma mutação do tipo “Best”, de maneira que o indivíduo de maior *fitness* é somado na mutação a uma taxa de 20%;
- PSO com 100 partículas, constantes $C1$ e $C2$ iguais a 2 e a constante ω de 0,8;

3.4 NORMALIZAÇÃO DOS DADOS

Antes dos dados serem inseridos no MLR é preciso realizar a normalização dos mesmos, uma vez que os valores de entrada apresentam magnitudes maiores que os valores dos limites da saída, o que pode gerar resultados incoerentes, mediante possibilidade tendenciosa da resposta ser o de maior valor possível dentro da sua respectiva função. Contudo, após a realização das previsões é necessário desnormalizar o valor estimado.

Os dados brutos (observados) informados pelo usuário passam pelo processo de normalização, em que o novo valor observado ($ObsNorm$) inserido no modelo foi

obtido através da Equação 13.

$$ObsNorm = \frac{Obs - media_{mes}}{variancia_{mes}} \tag{13}$$

na qual,

- $ObsNorm$: valor observado após o processo de normalização;
- Obs: valor observado;
- $media_{mes}$: média dos valores observados no respectivo mês da amostra observada;
- $variancia_{mes}$: variância obtida no mês.

Ainda, após obter os valores previstos, é necessário desnormalizar os valores obtidos utilizando a Equação 14.

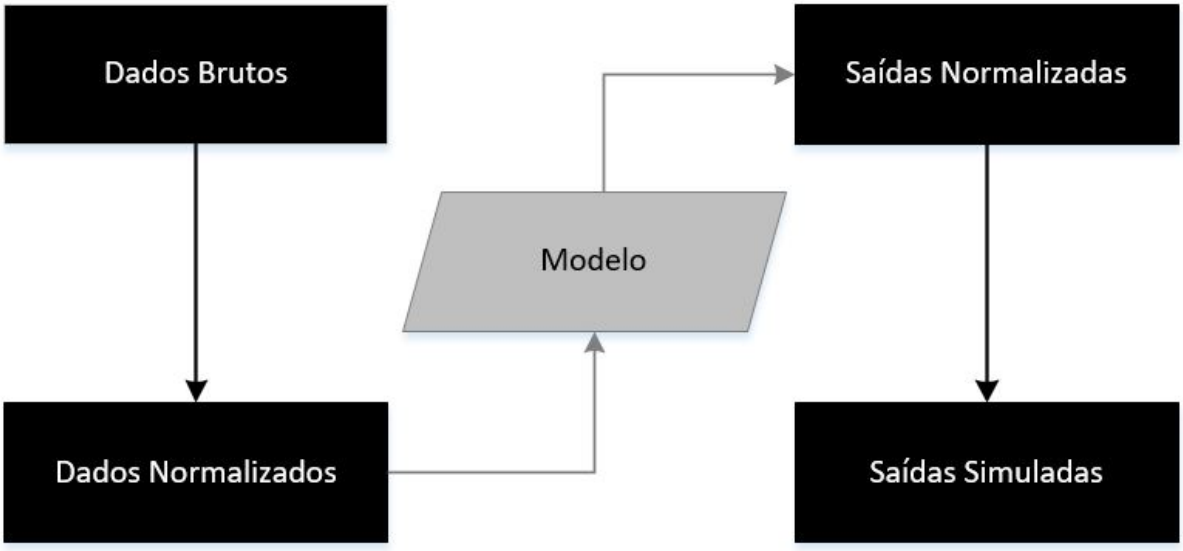
$$saidasimulada = saidapadronizada \times variancia_{mes} + media_{mes} \tag{14}$$

sendo,

- $saidasimulada$: valor previsto ou simulado;
- $saidapadronizada$: valor simulado normalizado;
- $media_{mes}$: média dos valores observados no respectivo mês da amostra;
- $variancia_{mes}$: variância obtida no mês.

A Figura 7 apresenta o fluxo da informação dentro da *framework* considerando a etapa supracitada.

Figura 7 – Layout do programa Qt Designer 5



Fonte: Autoria própria.

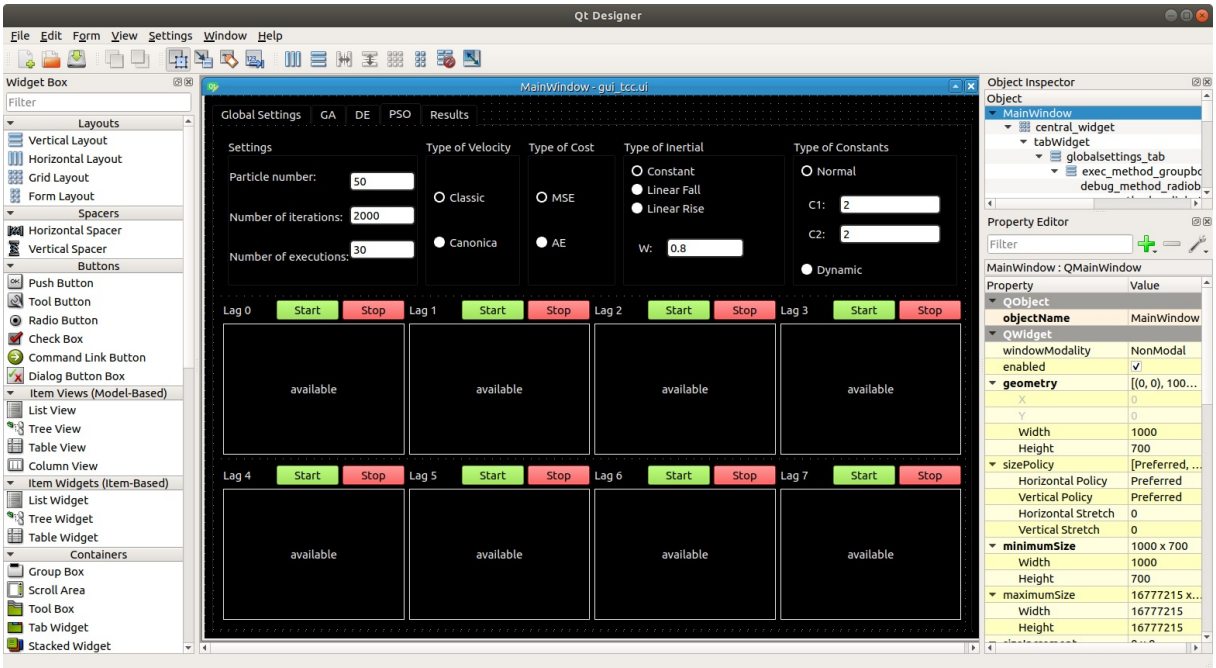
3.5 DESENVOLVIMENTO DA INTERFACE GRÁFICA DO *FRAMEWORK*

Para o desenvolvimento da interface gráfica buscou-se elaborar um *framework* intuitivo, com elementos visuais separados em grupos que facilitam ao usuário a configuração e execução das simulações. Como mencionado, optou-se pela elaboração do mesmo em uma ferramenta presente na gama de pacotes da biblioteca PyQt5, denominada Qt Designer 5 (PYQT, 2020).

Esta ferramenta auxilia na elaboração de uma Interface Gráfica de Usuário, do inglês *Graphical User Interface* (GUI) de maneira bastante visual, possibilitando a adição dos elementos desejados sem a necessidade de edição direta por código, proporcionando uma facilidade ao permitir ajustes finos nas configurações para cada elemento.

A Figura 8 mostra o *layout* do programa em que são apresentadas quatro partes principais: a *Widget Box*, utilizada para adicionar elementos a GUI da interface a ser criada; a área de *layout*, na qual se posicionam os objetos; o *object inspector*, que lista os nomes dos elementos adicionados, permitindo uma visualização simplificada do código a ser gerado; e o *Property Editor*, onde se realizam as customizações de cada objeto como formato, cor, funções, etc.

Figura 8 – Layout do programa Qt Designer 5



Fonte: Autoria própria.

Visando uma abrangência maior de usuários, optou-se por elaborar o programa

inteiramente na língua inglesa.

3.5.1 Abas Principais

Conforme citado nos itens da Seção 3.2 e da Seção 3.3, diversas funções e técnicas de otimização precisariam ser acrescentadas ao *framework* para possibilitar uma execução personalizada por parte do usuário de uma maneira simples e descomplicada. Assim o *layout* foi dividido em abas que apresentam as principais funções a serem determinadas em cada execução, além de uma específica para tratamento dos dados resultantes. Conforme mostra a Figura 9, dividiu-se a interface em cinco abas: *Global Settings*, GA, DE, PSO e *Results*.

Figura 9 – Abas principais

Fonte: Autoria própria.

3.5.2 Aba Configurações Globais

As características presentes na aba *Global Settings* são aquelas consideradas relevantes para todo o projeto a ser executado. Portanto, para evitar repetir as mesmas opções diversas vezes em cada aba, optou-se por criar uma com estas definições globais.

O primeiro elemento presente é a barra de carregamento do arquivo de dados de entrada a serem analisados, mostrado a seguir na Figura 10. Seu funcionamento se dá através do botão *Open*, posicionado a direita. Ao ser clicado, abre-se uma janela do explorador de arquivos e permite-se a seleção de um arquivo com formato de planilha com os dados que serão carregados no *framework*.

Figura 10 – Campo de carregamento de arquivo de dados

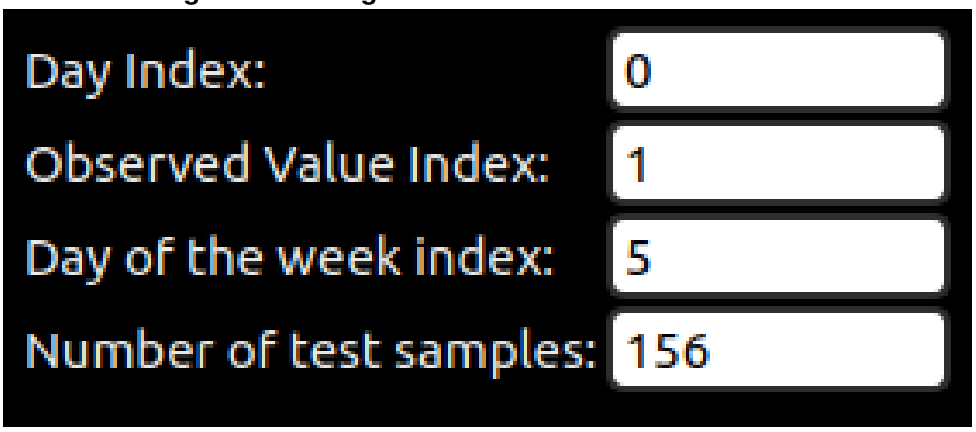
Fonte: Autoria própria.

As extensões de arquivo passíveis de serem carregadas são aquelas com formato .xls, xlsm e .ods por serem as mais comumente utilizadas pelos programas

corriqueiros de edição de planilhas, o Microsoft Office Excel e o LibreOffice Calc.

Logo abaixo posicionou-se quatro campos para inserção de dados que serão utilizados pelos algoritmos GA, DE e PSO. Os índices de Dia (*Day Index*), de valor observado (*Observed Value Index*) e de Dia da Semana (*Day of the week Index*), além do Número de Amostras para Teste (*Number of test samples*) foram posicionados verticalmente para melhor utilização do espaço. Os valores definidos que aparecem na Figura 11 foram os especificados previamente, podendo ser alterados de acordo com a vontade do usuário.

Figura 11 – Parâmetros globais dos algoritmos



Fonte: Autoria própria.

A Figura 12 apresenta um exemplo de uma tabela de base de entrada, na qual na primeira coluna (A) temos a data onde foi observada a ocorrência referente ao *Day Index*. Já a segunda coluna (B), representa nesse caso o número de internações por doenças respiratórias, representada pelo índice *Observed Value Index*.

Entre as colunas (C) e (E) temos os índices de material particulado (MP10), temperatura média (Temp) e umidade relativa do ar (UR). Na coluna (F) temos a representação binária se o dia é feriado ou não. Por fim temos a coluna (G), representando o dia da semana (*Day of the week Index*). Se o usuário desejar alterar a coluna onde estão os dados na tabela de entrada, basta informar na tela de parâmetros iniciais (Figura 11) estes índices.

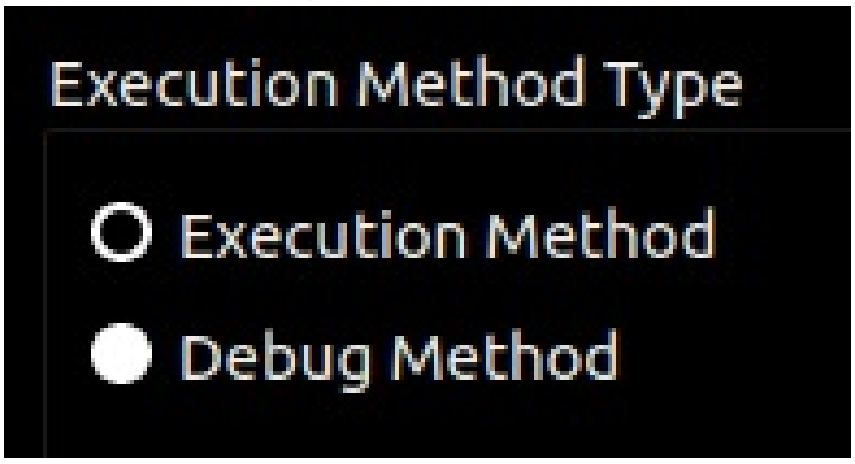
Figura 12 – Base de dados da cidade de São Paulo

	A	B	C	D	E	F	G
1	Data	IR	PM10	Temp	UR	Feriado	DiaSem
2	8/1/2014	117	18	27,2	33	0	3
3	9/1/2014	139	16	24,18	31	0	4
4	10/1/2014	112	21	24,76	30	0	5
5	11/1/2014	82	15	26,72	43	0	6
6	12/1/2014	93	14	23,2	38	0	7
7	13/1/2014	118	30	22,34	52	0	1
8	14/1/2014	139	31	22,86	38	0	2
9	15/1/2014	122	35	23,6	33	0	3
10	16/1/2014	84	27	22,8	55	0	4
11	17/1/2014	73	30	22,16	48	0	5

Fonte: Autoria própria.

Também foi inclusa uma opção para o tipo de método de execução (*Execution Method Type*). Caso deseje-se executar o *framework* em modo de depuração para encontrar possíveis erros no *software* deve-se utilizar o *Debug Method*. Já para a execução completa utiliza-se o *Execution Method*. A Figura 13 abaixo ilustra esse elemento.

Figura 13 – Tipos de método de execução



Fonte: Autoria própria.

Por fim, decidiu-se acrescentar à esta tela principal o logotipo do Laboratório de Inteligência Computacional e Controle Avançado (LICON), da UTFPR-PG, devido a sua importância no desenvolvimento deste trabalho, bem como uma homenagem aos acadêmicos que dele fazem parte e que de alguma forma contribuíram para a realização do mesmo. A Figura 14 mostra a tela completa da aba *Global Settings* do programa.

Figura 14 – Aba *Global Settings*

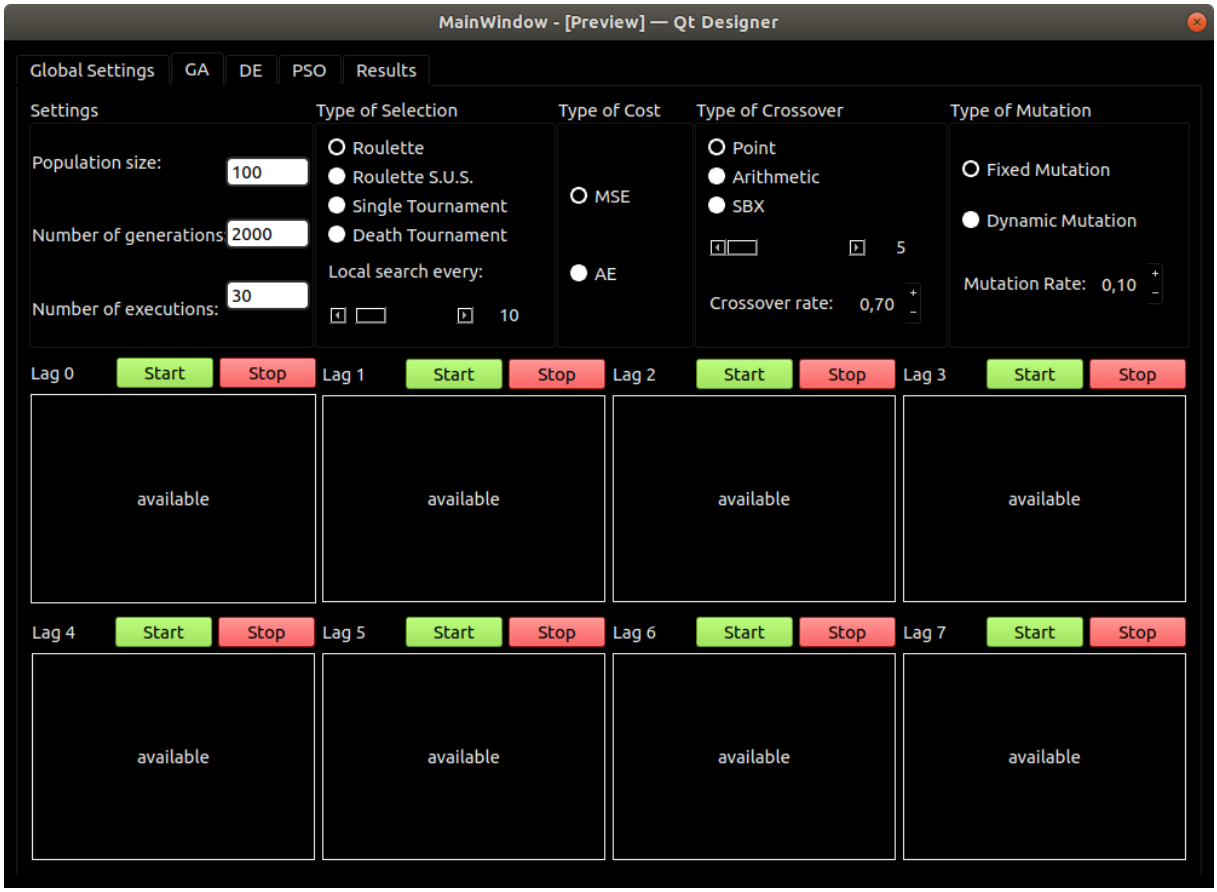
Fonte: Autoria própria.

3.5.3 Aba Algoritmo Genético (GA)

Na aba seguinte constam as configurações referentes à execução do algoritmo genético, apresentado na Seção 2.2.1. Os elementos estão dispostos em duas partes: na primeira constam as características do algoritmo e seus parâmetros e na segunda foram dispostos oito campos para execução do *lag* 0 ao *lag* 7.

Abaixo destes elementos foram colocados os oito campos para os *lags*, e um botão de início e de parada para cada dia, permitindo ao usuário optar pela melhor forma de execução de acordo com a capacidade de cada equipamento que este dispõe. A Figura 15 mostra o *layout* final.

Figura 15 – Aba do Algoritmo Genético



Fonte: Autoria própria.

As características configuráveis deste algoritmo envolveram: Opções (tamanho da população, número de gerações, número de execuções, taxa de mutação, taxa de *crossover*; Tipo de custo e busca local a cada × intervalo de gerações); Tipo de seleção (roleta, roleta s.u.s, torneio simples ou torneio da morte); Tipo de *crossover* (ponto, aritmético, SBX) e, Tipo de mutação (mutação fixa, mutação dinâmica). Essas características são descritas com maiores detalhes nas subseções a seguir.

3.5.3.1 Opções

- **Tamanho da população:** Número total de indivíduos da população.
- **Número de gerações:** Quantidade de vezes que serão executados o processo de *crossover*, seleção e mutação dentro da população.
- **Número de execuções:** Número de vezes que o algoritmo será executado.
- **Taxa de *crossover*:** Probabilidade dos indivíduos da população realizarem *crossover*.

- **Taxa de mutação:** Taxa da população em que é executada a mutação em cada iteração.
- **Função custo:** Erro Quadrático Médio (MSE) ou Erro Absoluto (AE).
- **Busca local:** Realiza uma busca local a cada n gerações. No *framework* proposto, ela realiza uma busca local do tipo "subida/descida". Esta técnica consiste na varredura dos genes, em que é somado um número randômico b ao cromossomo.; Após é verificado se o valor do *fitness* obteve melhores resultados. Caso contrário, é encerrada a busca local de subida. Em seguida, inicia-se a busca local de descida, na qual é subtraído o valor randômico b do cromossomo. Enquanto o *fitness* melhora esse processo de subtração este processo se repetido. Caso contrário, é encerrado o processo.

3.5.3.2 Tipos de Seleção

- **Roleta:** A probabilidade de que o indivíduo seja escolhido é proporcional a magnitude de seu *fitness*.
- **Roleta S.U.S.:** Nesta tipo de seleção a magnitude do *fitness* não aumenta as chances do indivíduo ser escolhido.
- **Torneio Simples:** Neste torneio são selecionados dois indivíduos aleatórios da população, de modo que o indivíduo com o maior *fitness* permanece na população.
- **Torneio da morte:** No torneio da morte, os indivíduos se enfrentam aos pares e o indivíduo de maior *fitness* continua na população enquanto o perdedor é suprimido. Nota-se que em geral esta proposta apresenta menor diversidade.

3.5.3.3 Tipos de *Crossover*

- **Ponto:** Este tipo de *crossover* é apresentado na Figura 2.
- **Aritmético:** Neste caso, inicialmente realiza-se o *crossover* de ponto e após são gerados um número aleatório $beta$ e um número $alfa$ que é definido pela Equação 15.

$$alfa = 1 - beta \tag{15}$$

Se o número do gene for menor que o ponto, o cromossomo é multiplicado pelo número $beta$. Senão, o cromossomo é multiplicado por $alfa$.

- **SBX:** No *crossover* SBX, o usuário configura um número positivo $MSBX$ e, em seguida, são escolhidos dois pais aleatórios dentro da população. Um número randômico (u) entre 0 e 1 é selecionado. Se o número randômico for menor que 0.5, um coeficiente $beta$ é obtido através da Equação 16.

$$beta = (2u)^{1/MSBX+1} \tag{16}$$

Caso contrário, o coeficiente $beta$ é definido conforme a Equação 17.

$$beta = (0.5/1-u)^{1/MSBX+1} \tag{17}$$

Por fim, para cada cromossomo i do $filho1$ utiliza-se a equação Equação 18:

$$filho1.cromo_i = 0.5 \times (pai1_i + pai2_i) - 0.5 \times beta \times (pai1_i - pai2_i) \tag{18}$$

De forma similar, para cada cromossomo i do $filho2$ utiliza-se a Equação 19.

$$filho2.cromo_i = 0.5 \times (pai1_i + pai2_i) + 0.5 \times beta \times (pai1_i - pai2_i) \tag{19}$$

3.5.3.4 Tipos de Mutação

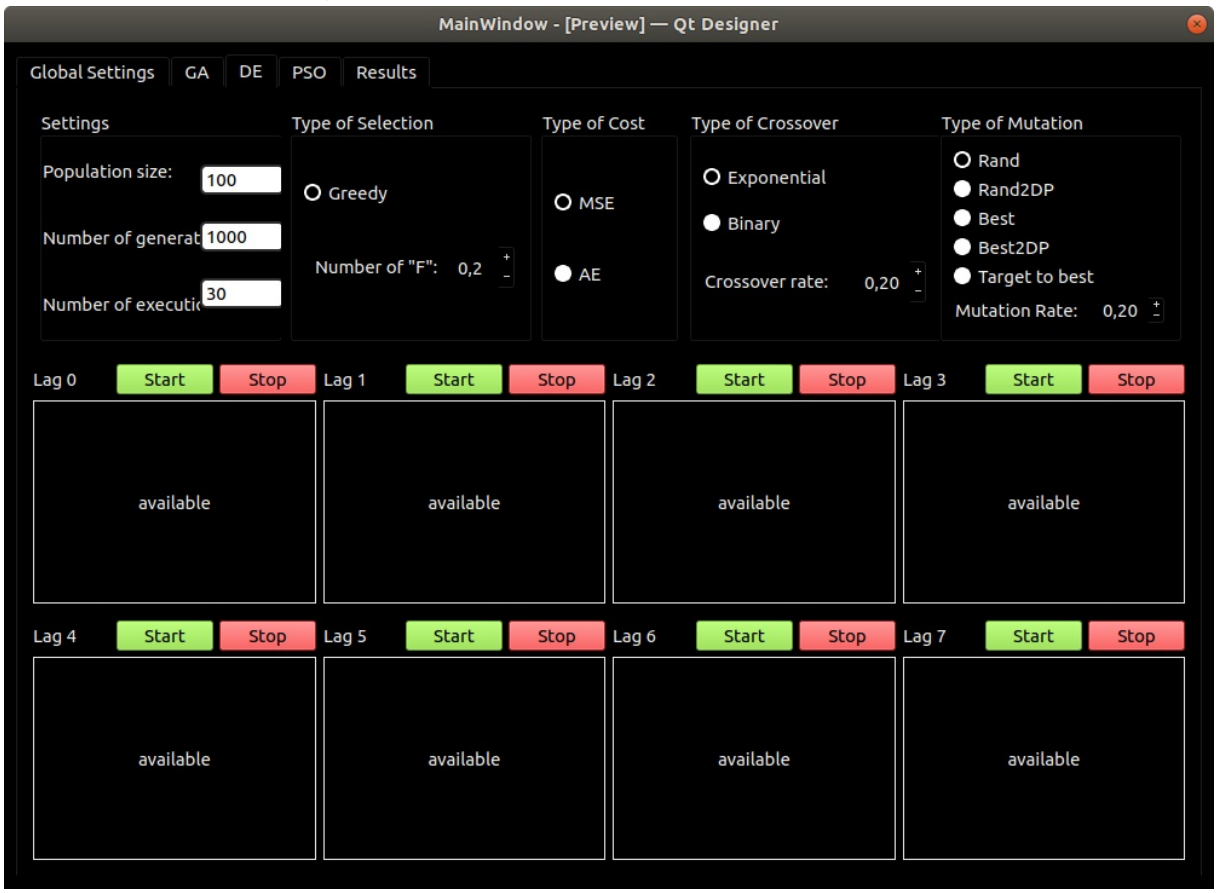
- **Mutação fixa:** No *framework* são gerados novos coeficientes para a taxa de mutação $taxa_{mut}$ aleatórios entre -1 e 1, para a porcentagem da população definida pelo usuário.
- **Mutação dinâmica:** A mutação dinâmica a taxa de mutação ($taxa_{mut}$) possui uma relação direta com o *fitness* do melhor indivíduo, em que seu valor é acrescido geração após geração. Deste modo, a cada geração, a taxa de mutação é recalculada através da Equação 20.

$$taxa_{mut} = 30 \times (1 - ((1 - \frac{\sum_1^{geracao} gbest.fit}{tam_{populacao}})/1)) \tag{20}$$

3.5.4 Aba Evolução Diferencial (DE)

Para a aba DE do *framework*, foi mantido o *layout* anterior alterando apenas as funções características desse algoritmo, conforme discutido previamente na Seção 2.2.2. Novamente utilizou-se os campos individuais para cada *lag*, como pode ser observado na Figura 16.

Figura 16 – Aba da Evolução Diferencial



Fonte: Autoria própria.

As características configuráveis deste algoritmo envolvem: Opções (tamanho da população, Número de Gerações, Número de Execuções, Número "F", Taxa de Mutação, Taxa de *crossover* e Tipo de Custo), Tipo de Seleção (Greedy), Tipo de *crossover* (exponencial ou binário) e por fim, Tipo de Mutação (*Rand*, *Rand2DP*, *Best*, *Best2DP* ou *Target to Best*). Descritos com maiores detalhes nas subseções a seguir.

3.5.4.1 Opções

- **Tamanho da população:** Número total de indivíduos da população.

- **Número de gerações:** Quantidade de vezes que serão executados o processo de *crossover*, seleção e mutação dentro da população.
- **Número de execuções:** Número de vezes que o algoritmo será executado.
- **Taxa de *crossover*:** Probabilidade dos vetores (agentes) da população realizarem *crossover*.
- **Taxa de mutação:** Taxa da população em que é executa a mutação em cada iteração.
- **Função custo:** Erro Quadrático Médio (MSE) ou Erro Absoluto (AE).
- **Número "F":** Realiza o ajuste do universo de discurso.

3.5.4.2 Tipos de Seleção

- **Greedy:** Na seleção de Greedy se o *fitness* do *Trial Vector* for melhor que o *fitness* atual do indivíduo, o *Trial Vector* substitui o anterior na população de vetores.

3.5.4.3 Tipos de *Crossover*

- **Exponencial:** Inicialmente é selecionado um ponto aleatório entre 0 e o número de vetores. Enquanto o número de coeficientes for menor que o número do ponto, é executado uma verificação. Se a taxa de *crossover* for maior ou igual a um número randômico gerado entre 0 e 1, o cromossomo do *Trial Vector* recebe o cromossomo do *Donor Vector* e ao ponto é acrescido mais 1. Senão, o processo iterativo é interrompido.
- **Binário:** No *crossover* binário do experimento de Bernolli, um número randômico é gerado entre 0 e 1. Se a taxa de *crossover* for maior ou igual ao número aleatório gerado, o gene do *Trial Vector* passa a ser o do do *Donor Vector*. Caso contrário, o gene do *Trial Vector* passa para o vetor do próprio indivíduo.

3.5.4.4 Tipos de Mutação

- ***Rand*:** São escolhidos 3 indivíduos diferentes ($t1$, $t2$ e $t3$) aleatoriamente dentro da população. Como citado acima, a variável F ajusta o universo de discurso. No processo de Mutação surge o *Donor Vector* cujo cálculo é apresentado na Equação 21 para cada dimensão d do agente:

$$DonorVector_d = t1_d + F \times (t2_d - t3_d) \tag{21}$$

- ***Rand2DP*:** São escolhidos 5 indivíduos diferentes ($t1$, $t2$, $t3$, $t4$ e $t5$) aleatoriamente dentro da população. Assim, o cálculo do *Donor Vector* neste caso segue a Equação 22.

$$DonorVector_d = t1_d + F \times ((t2_d - t3_d) + (t4_d - t5_d)) \tag{22}$$

- ***Best*:** São escolhidos 2 indivíduos diferentes ($t2$ e $t3$) aleatoriamente dentro da população, em que o melhor indivíduo ($best$) é utilizado para a realização do processo de mutação. O *Donor Vector* é definido como na Equação 23.

$$DonorVector_d = best_d + F \times (t2_d - t3_d) \tag{23}$$

- ***Best2DP*:** São escolhidos 4 indivíduos diferentes ($t2$, $t3$, $t4$ e $t5$) aleatoriamente dentro da população, de modo que o melhor indivíduo ($best$) é utilizado para a realização do processo de mutação. A Equação 25 é então utilizada:

$$DonorVector_d = best_d + F \times ((t2_d - t3_d) + (t4_d - t5_d)) \tag{24}$$

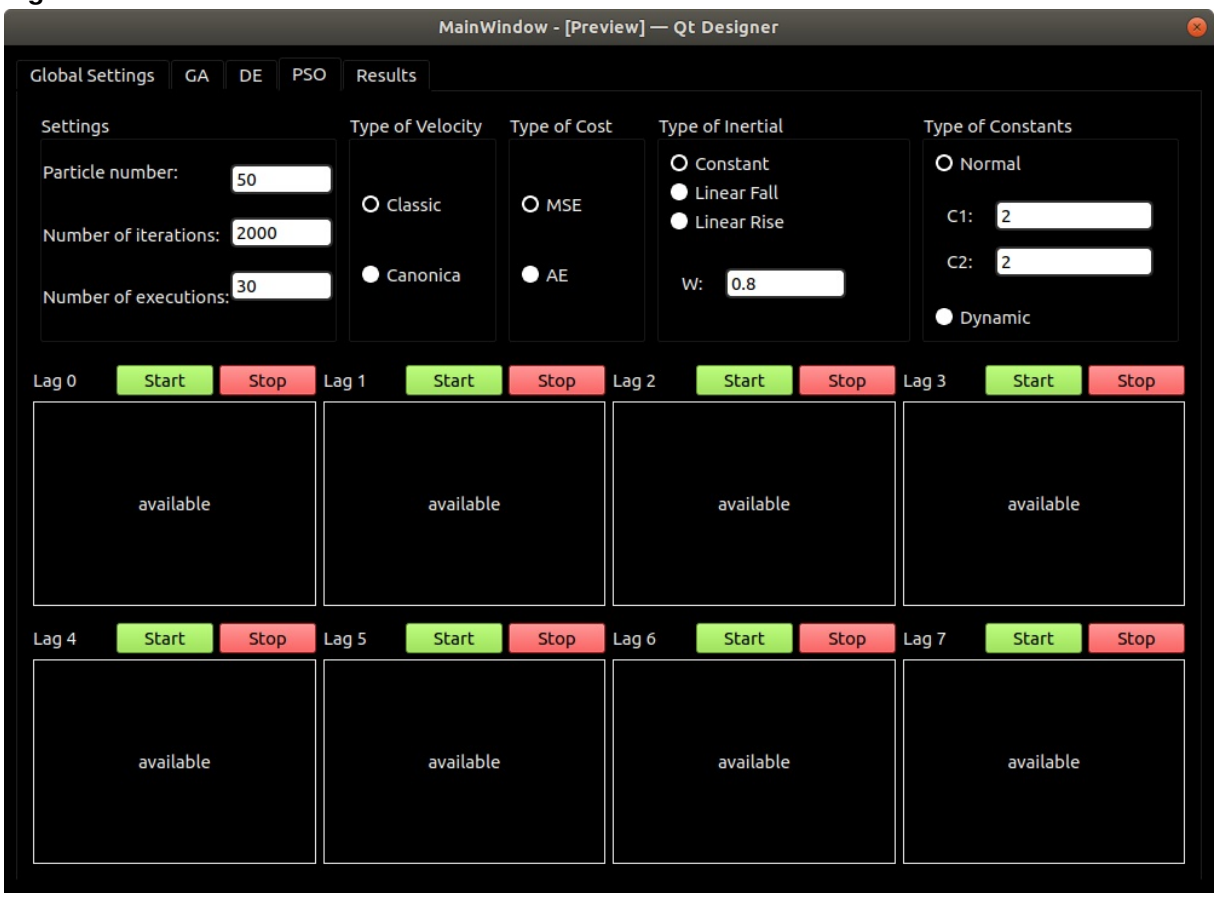
- ***Target to Best*:** São escolhidos 2 indivíduos diferentes ($t1$ e $t3$) aleatoriamente dentro da população, e ainda, o melhor indivíduo ($best$) é utilizado para a realização do processo de mutação. O cálculo do *Donor Vector* é como na Equação 25.

$$DonorVector_d = t1_d + F \times (best_d - t3_d) \tag{25}$$

3.5.5 Aba Otimização por Enxame de Partículas (PSO)

O PSO possui uma forma de execução diferente das apresentadas anteriormente como apresentado na Figura 17.

Figura 17 – Aba do PSO



Fonte: Autoria própria.

A Figura 17 apresenta a disposição final dos elementos e dos valores adotados para a realização dos testes, cujas características configuráveis envolvem: Opções (Número de partículas, Número de iterações, Número de execuções, Peso de inércia (ω), Constante social ($C2$), Constante cognitiva ($C1$), Tipo de custo); Tipo de velocidade (clássica ou canônica); Tipo inercial (constante, queda linear e aumento linear) e por fim, Tipo de constantes (normal ou dinâmica). Essas características são descritas com maiores detalhes nas subseções a seguir.

3.5.5.1 Opções

- **Número de partículas:** Número de partículas do enxame.

- **Número de iterações:** Quantas vezes serão recalculadas a velocidade e posição das partículas.
- **Número de execuções:** Número de vezes que o algoritmo será executado.
- **Peso de inércia (ω):** Normalmente definido com 0.8.
- **Constante Cognitiva ($C1$):** Normalmente definido com 2.
- **Constante Social ($C2$):** Normalmente definido com 2.
- **Tipo de custo:** Erro Quadrático Médio (MSE) ou Erro Absoluto (AE).

3.5.5.2 Tipos de velocidade

- **Clássica:** O cálculo da velocidade clássica foi apresentado na Equação 3.
- **Canônica:** Na velocidade canônica, uma contante χ é obtida através da Equação 26.

$$\chi = \frac{2 \times k}{\left| 2 - fi - \sqrt[2]{fi \times 2 - 4 \times fi} \right|} \tag{26}$$

Em que, k e fi são constantes, usualmente definidas como 1 e 10, respectivamente. Desta forma, para realizar o cálculo da velocidade canônica basta multiplicar a velocidade clássica por χ, como indicado na Equação 27.

$$velocidade = velocidadeclassica \times \chi \tag{27}$$

3.5.5.3 Tipos inerciais

- **Constante:** As constantes social, cognitiva e peso de inércia permanecem inalteradas ao longo do processo de otimização.
- **Queda linear:** Na queda linear, o peso de inércia (ω) diminui ao longo das iterações. Ainda, são definidos dois parâmetros no *framework*, ω_{sup} e ω_{inf}, com valores máximo (0,9) e mínimo (0,1). Estes são os valores que ω poderá atingir. A Equação 28 apresenta seu cálculo.

$$\omega = \omega_{sup} - ((\omega_{sup} - \omega_{inf}) / num_{iteracoes}) \times iteracao \tag{28}$$

3.5.5.4 Tipos de constantes

- **Normal:** $C1$ e $C2$ permanecem as informadas pelo usuário.
- **Dinâmica:** quando a opção tipo de contantes é selecionada, $C1$ e $C2$ se tornam variáveis ao longo das iterações. $C1$ é calculado por meio da Equação 29 e tem a constante inferior $C1_{inf}$ definida como 0,1.

$$C1 = C1 + ((C1_{inf} - C1)/num_{iteracoes}) \times iteracao \tag{29}$$

De forma similar, $C2$ é obtida através da Equação 30 e $C2_{inf}$ é definida como 2,5:

$$C2 = C2 + ((C2_{inf} - C2)/num_{iteracoes}) \times iteracao \tag{30}$$

Após a elaboração das abas de configurações globais e das especificidades para cada algoritmo, foi desenvolvida uma aba para o processamento dos dados resultantes conforme a execução do programa, almejando um tratamento dos dados de forma que facilite a interpretação dos resultados.

3.5.6 Aba de Resultados

Semelhante ao campo de carregamento de arquivo de dados, foi criado um campo para o *upload* dos arquivos resultantes do processo. Conforme mostra a Figura 18, um botão foi adicionado a direita e, após selecionado, o nome do arquivo é exibido no campo em branco. O tipo de arquivo aceito pelo programa é json gerado pelo *framework* após as simulações.

Figura 18 – Campo de carregamento dos dados resultantes

Fonte: Autoria própria.

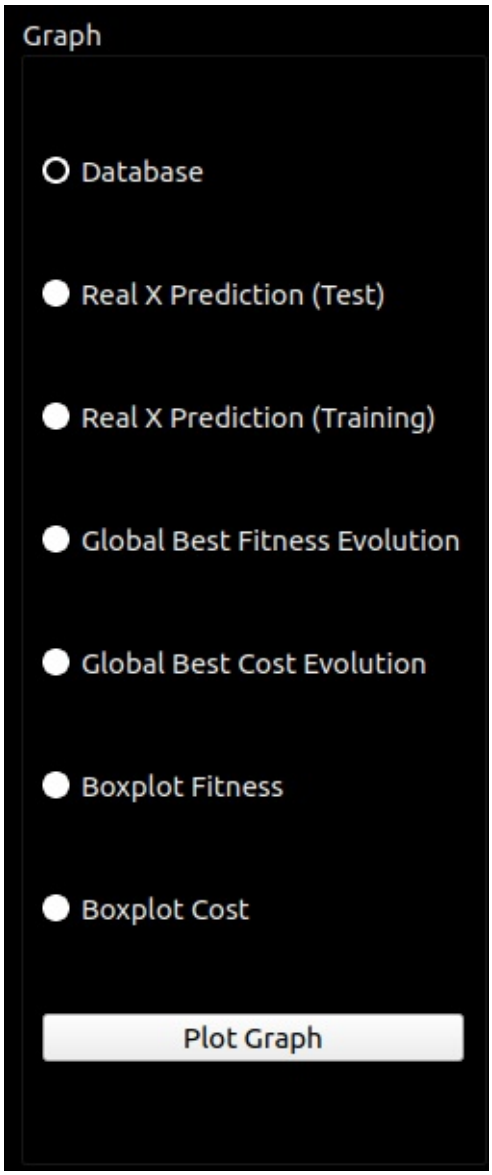
Para realizar a análise dos dados obtidos é necessária a plotagem de gráficos que permitam a interpretação visual dos valores. A inclusão de um *frame* com opções para geração dos gráficos dentro do próprio programa tornou-se necessária por ser uma etapa essencial no processo analítico.

As opções de plotagem são:

- Base de dados
- Real x predição (Teste)
- Real x predição (Treinamento)
- Evolução do melhor *fitness* global
- Evolução do melhor custo global
- Dispersão do *fitness*
- Dispersão do custo

Tais opções são apresentadas na Figura 19:

Figura 19 – Opções para plotagem de gráficos



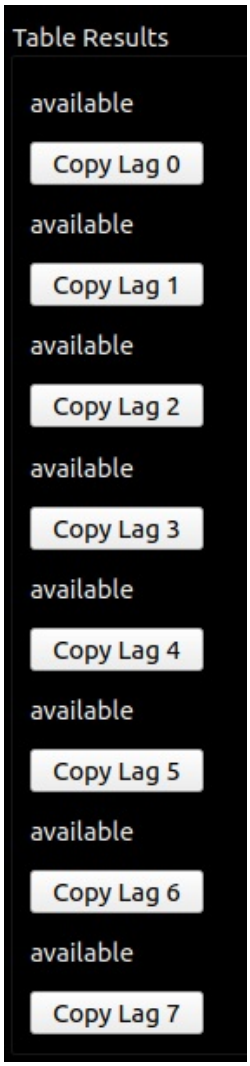
Fonte: Autoria própria.

Além dos gráficos, verificou-se a necessidade de aquisição de alguns dados específicos da execução. Os valores de AE, MSE, MAPE, ARV, IA, MAE, RMSE e

MEAN eram necessários para um estudo de desempenho de cada algoritmo. Assim, tais valores são calculados utilizando o próprio *framework*. Para que não fosse preciso buscar essas informações manualmente criou-se um outro *frame* para obtenção desses valores.

As informações foram organizadas para cada *lag* e um botão foi acrescentado com a função de copiar os valores já no formato correto para colagem em uma planilha. A Figura 20 mostra a tabela de resultados. Após o carregamento do arquivo os dados são calculados e substituídos nos rótulos “available”.

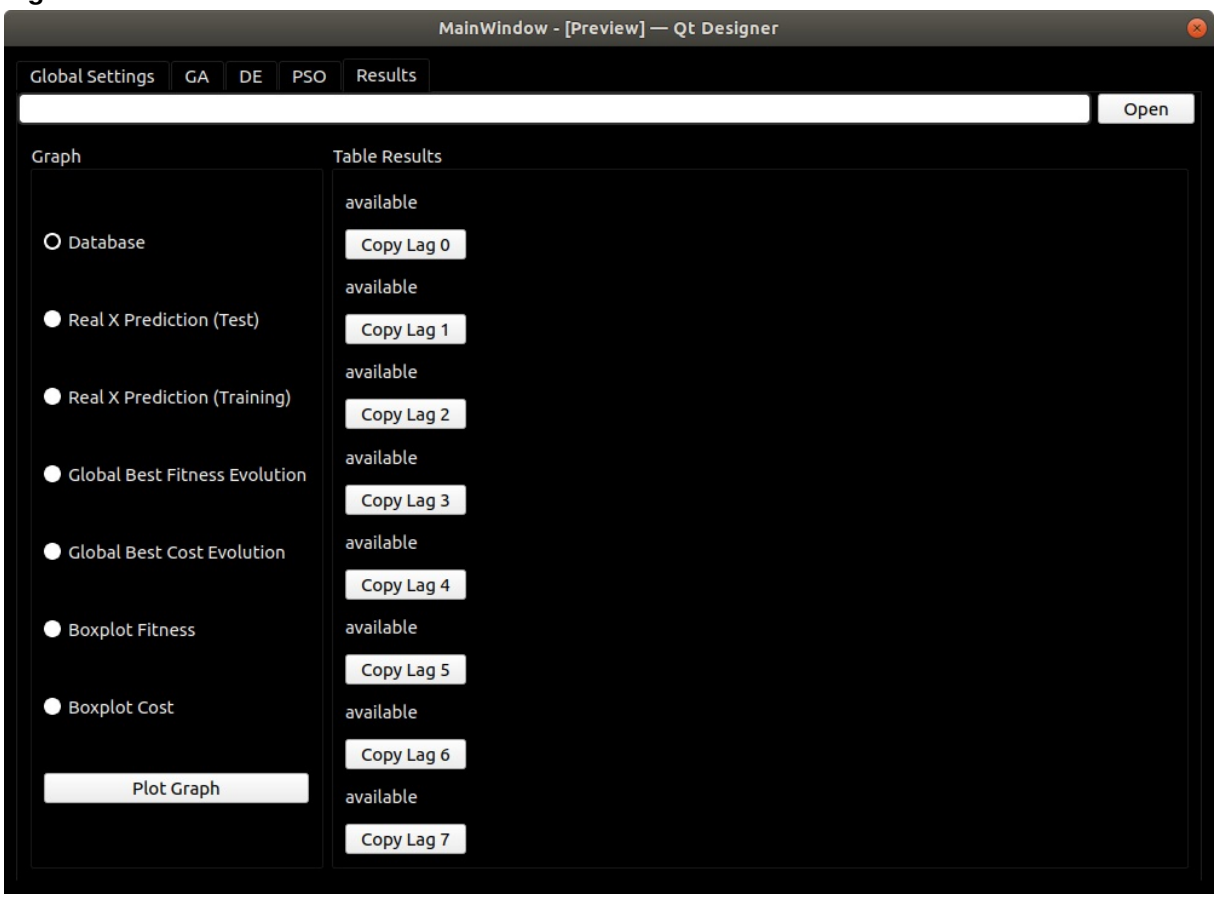
Figura 20 – Tabela de Resultados



Fonte: Autoria própria.

A Figura 21 ilustra a disposição final dos elementos no *layout* do *framework*. Apesar do programa permitir um ajuste de tela, reservou-se um espaço maior para a tabela de resultados para que todos os dados pudessem ser exibidos adequadamente.

Figura 21 – Aba *Results*



Fonte: Autoria própria.

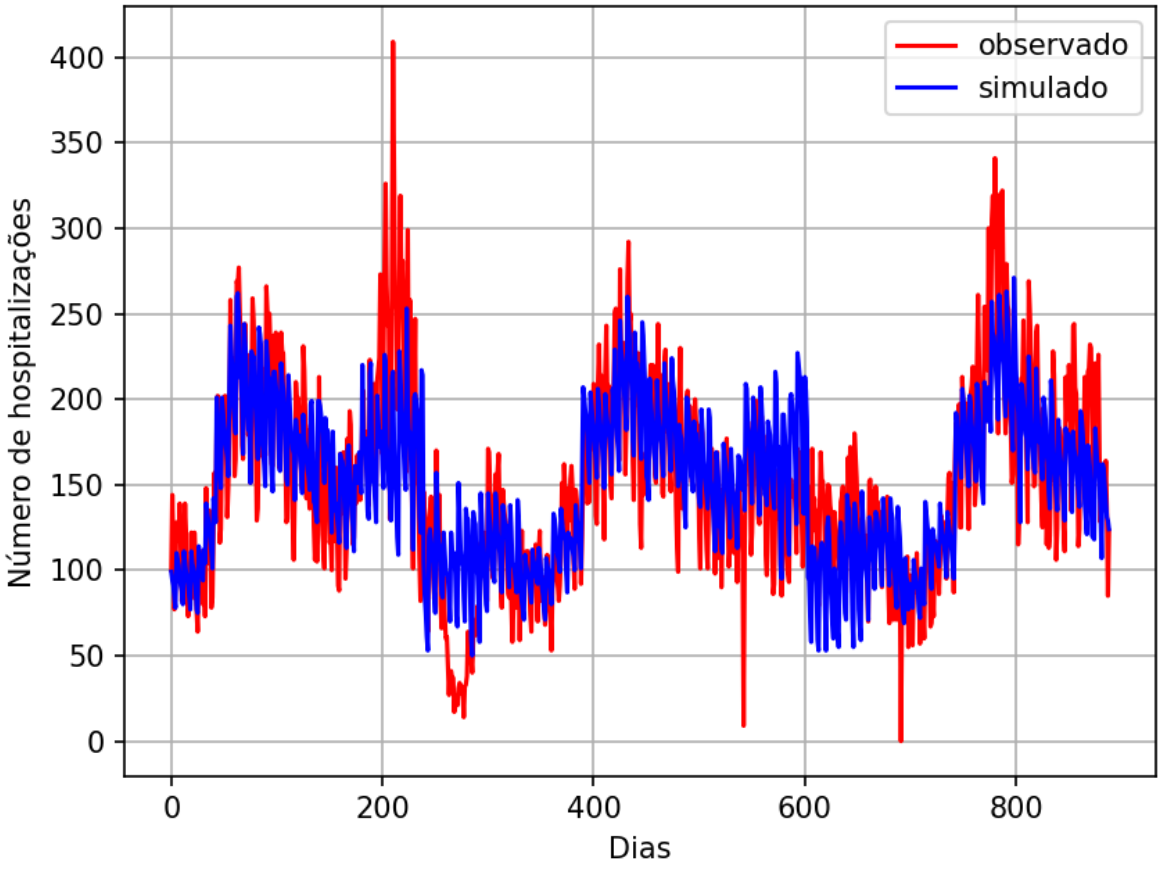
4 RESULTADOS E DISCUSSÃO

Este capítulo apresentando os resultados e as respectivas discussões sobre o presente estudo. As configurações livres dos modelos de otimização são as mencionadas na Seção 3.3.

4.1 REGRESSÃO LINEAR MÚLTIPLA OTIMIZADA PELA EVOLUÇÃO DIFERENCIAL (MLR-DE)

Esta seção apresenta os resultados obtidos mediante o uso do MLR calibrado pelo algoritmo DE. A Figura 22 apresenta duas curvas de treinamentos, em vermelho referente aos valores de internações hospitalares observados, enquanto a curva em azul refere-se aos valores obtidos pelo modelo MLR-DE.

Figura 22 – Curvas de Treinamento entre as variáveis Simuladas e Observadas

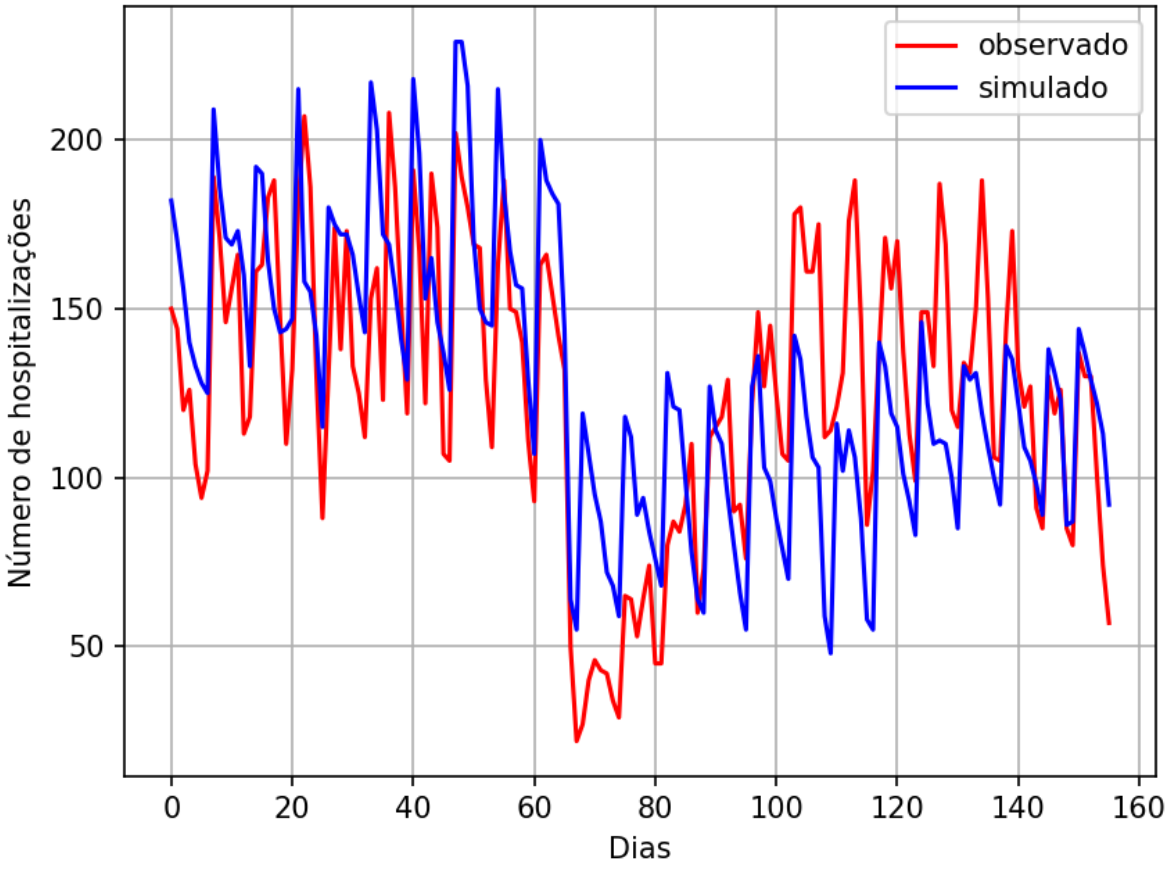


Fonte: Autoria própria.

Ao analisar a Figura 22 é possível observar que o modelo teve maior dificuldade em representar a curva observada por volta dos duzentos dias, apresentando assim, um maior erro neste período.

A seguir, a Figura 23 apresenta as curvas de teste entre as duas variáveis em estudo, observado e simulados.

Figura 23 – Curvas de Teste entre as variáveis Simuladas e Observadas

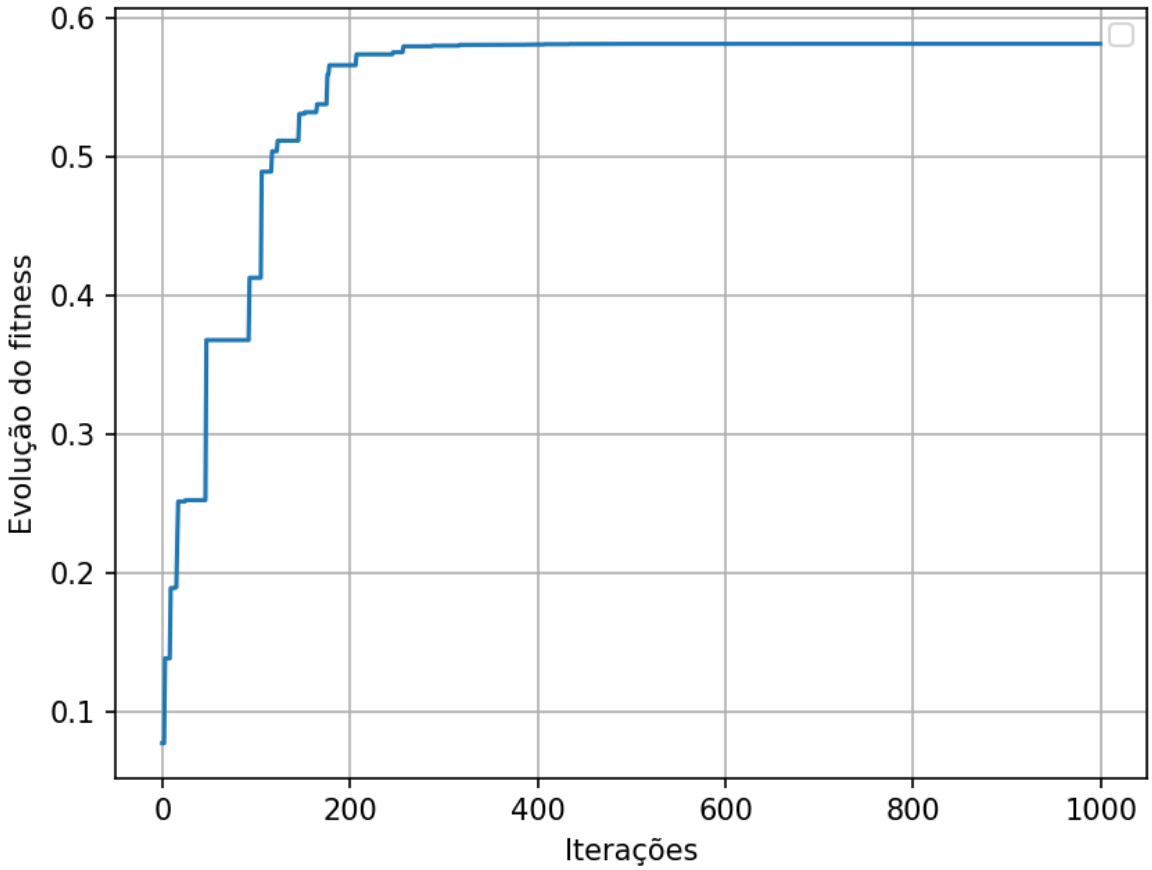


Fonte: Autoria própria.

Ao analisar a Figura 23 observa-se que os valores observados e simulados foram maiores entre 60 e 80 dias. Porém, entre 100 e 140 dias, houve um período longo em que os valores da curva simulado ficaram abaixo dos valores observados.

A Figura 24 apresenta a curva de evolução do fitness MSE para o melhor indivíduo.

Figura 24 – Evolução do Fitness para o Melhor Indivíduo

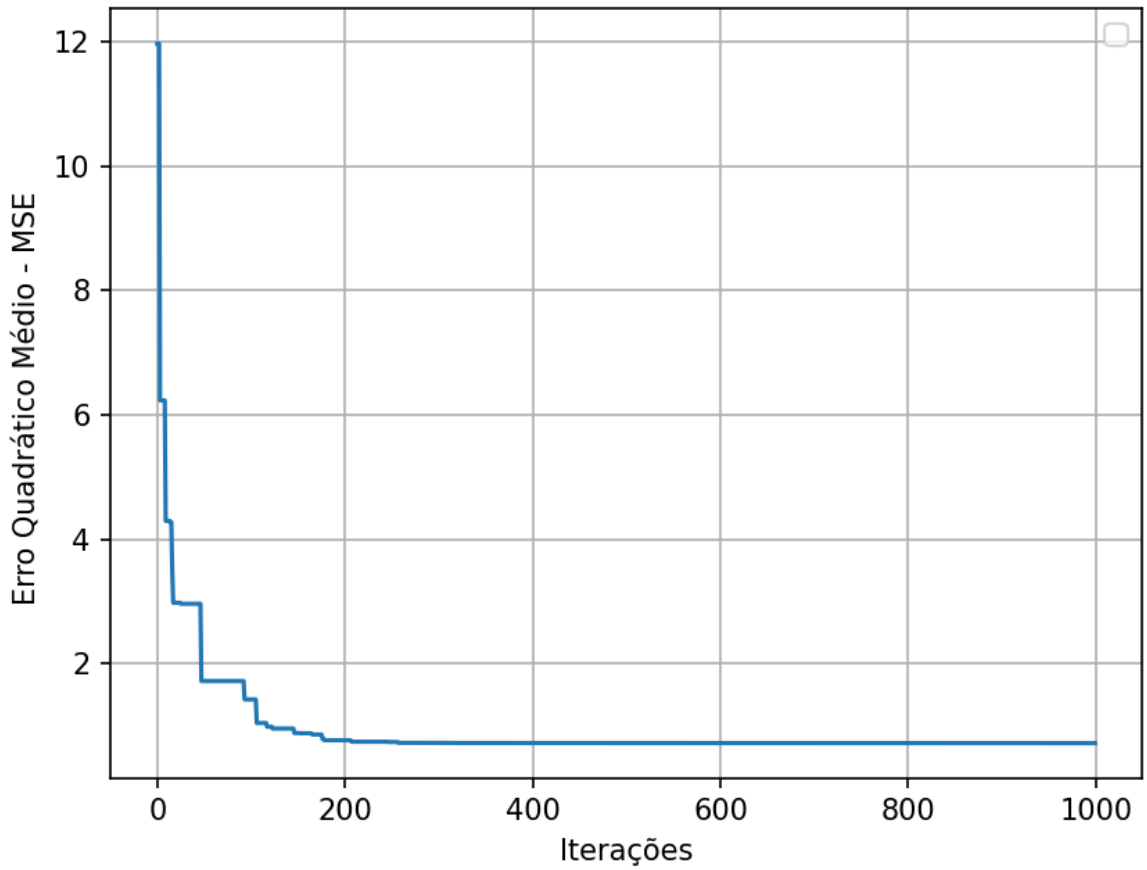


Fonte: Autoria própria.

A Figura 24 aponta a inicialização do algoritmo com um *fitness* de aproximadamente 0,05, sendo que até atingir 200 iterações, houve uma rápida evolução, quando o *fitness* estabilizou em pouco mais de 0,55. Ainda, observa-se que em 1000 interações, houve a convergência do algoritmo.

A Figura 25 apresenta a curva do custo (MSE) para o melhor indivíduo.

Figura 25 – Evolução do Custo para o Melhor Indivíduo

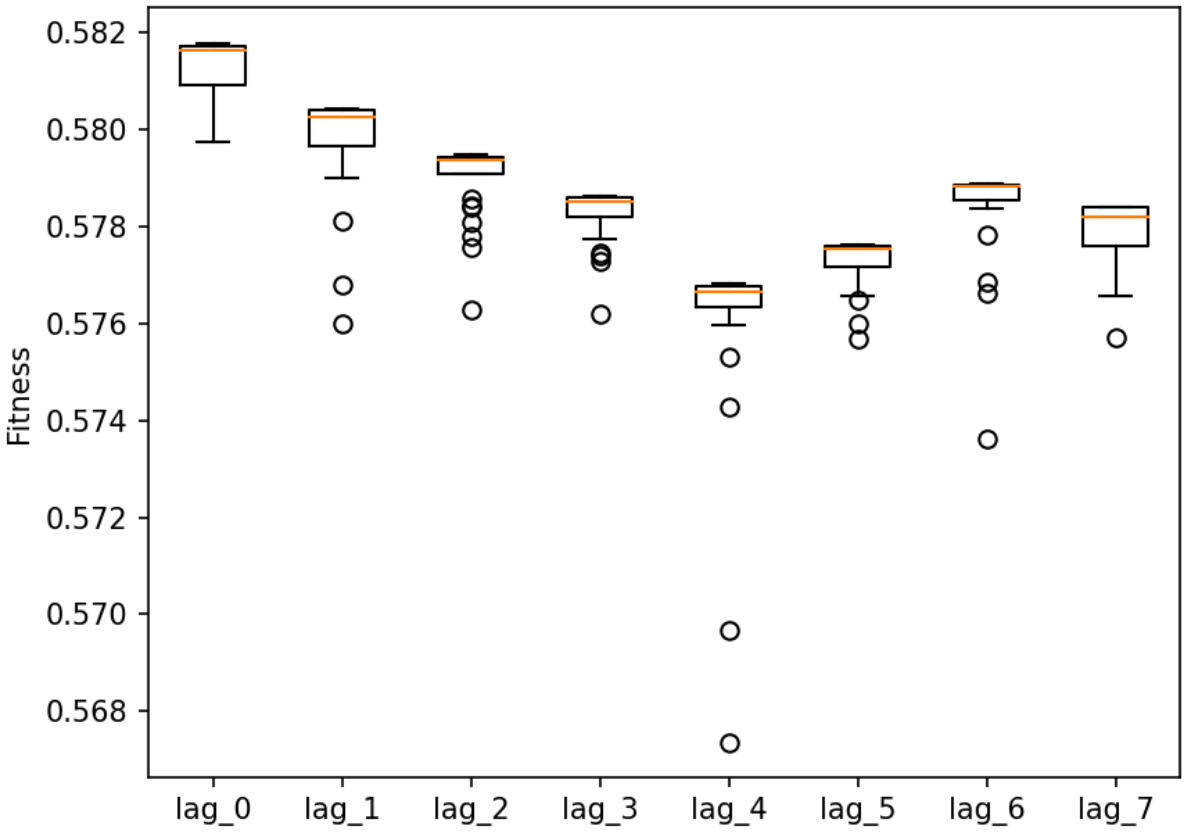


Fonte: Autoria própria.

Observa-se na Figura 25 que a solução inicial apresentou um Erro Quadrático Médio (MSE) inicial de 12. Após aproximadamente 200 iterações o MSE passou a ser menor que 1.

A Figura 26 apresenta a dispersão do *fitness* (que deve ser maximizado) para os 8 *lags*, em que é possível observar que para o lag 0, houve o melhor desempenho, em aproximadamente 0,582. Já para o *lag* 4, houve o pior *fitness*.

Figura 26 – Dispersão do Fitness para os *lags*

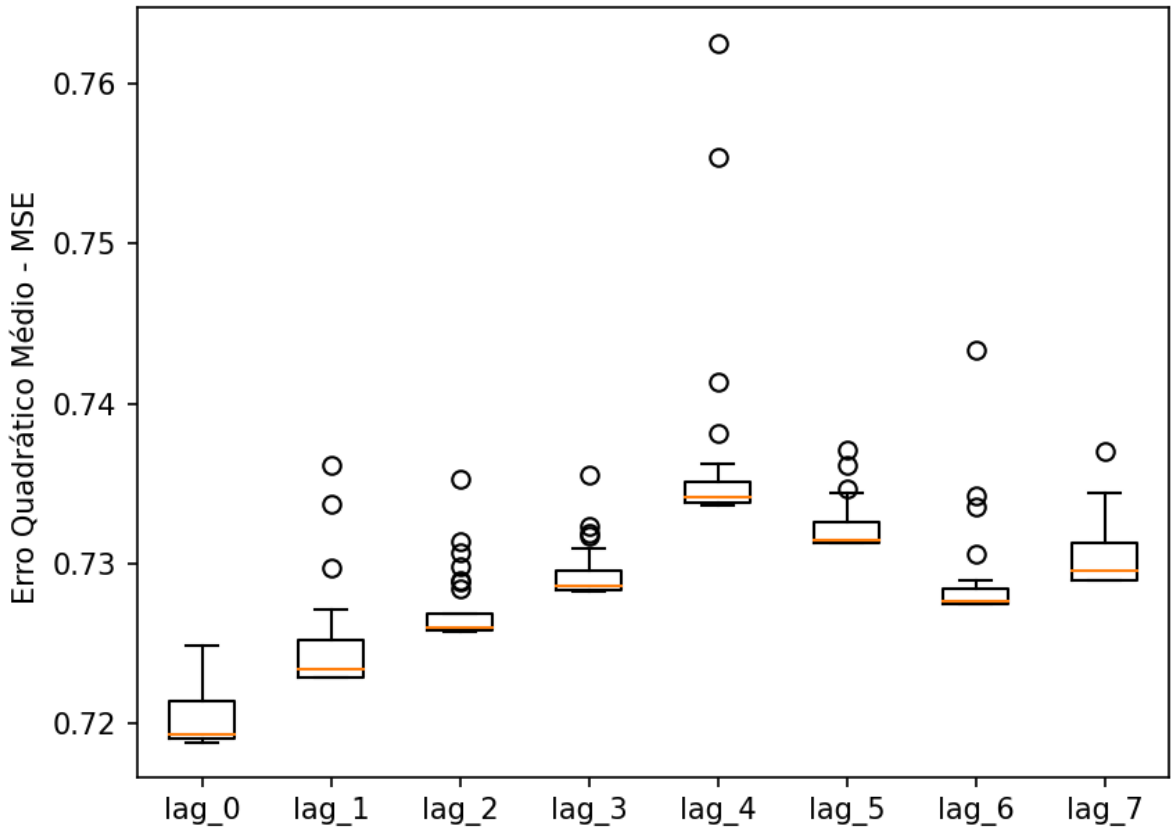


Fonte: Autoria própria.

Ainda, mediante análise da Figura 26, é possível observar que o grau de dispersão entre os valores encontrados foi baixo, pois há uma baixa oscilação entre as médias dos resultados obtidos.

Por fim, a Figura 27 apresenta o gráfico de dispersão MSE para 8 *lags*, sendo possível observar que para o *lag* 4 houve o maior erro, de aproximadamente 0,76 que segue o comportamento do *fitness*, já que são inversamente proporcionais. Ainda, a Figura 27 aponta que o menor erro foi para o *lag* 0.

Figura 27 – Dispersão do Custo para 8 *lags*

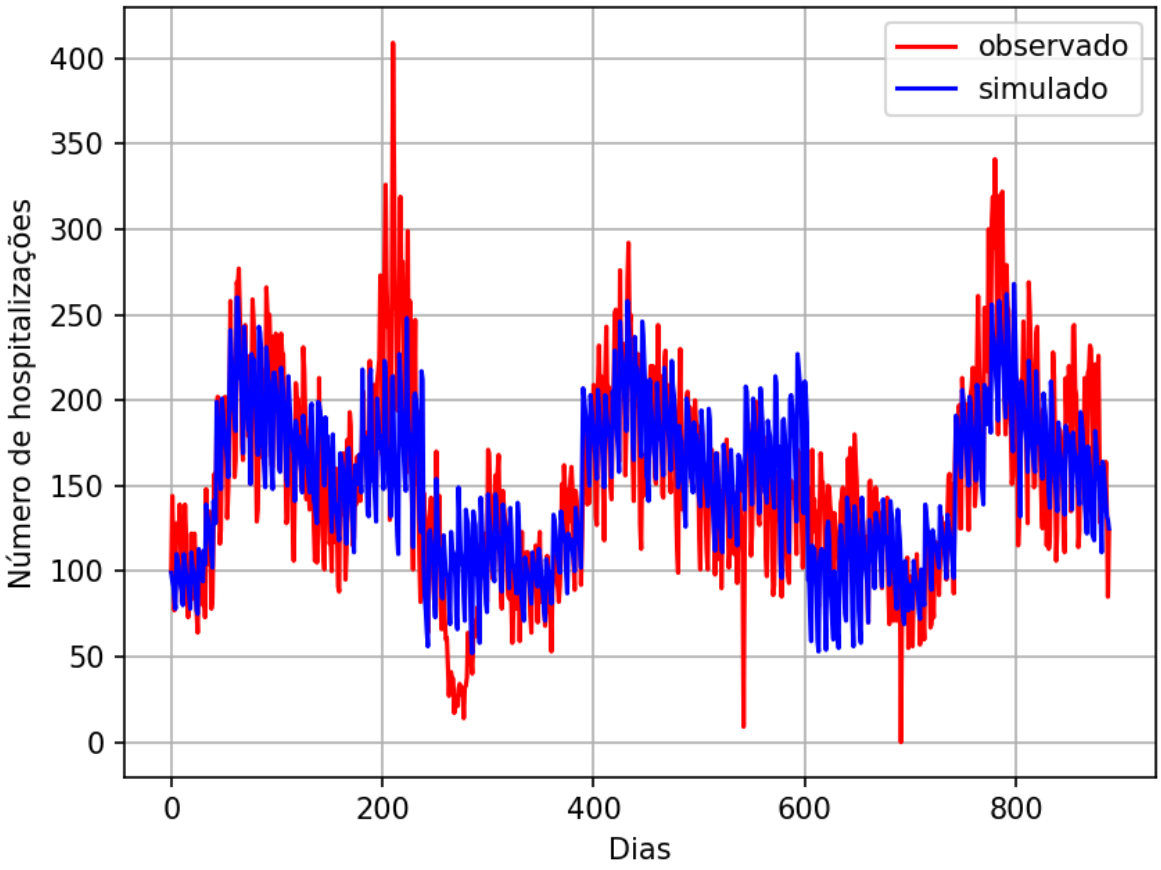


Fonte: Autoria própria.

4.2 REGRESSÃO LINEAR MÚLTIPLA OTIMIZADA PELO ALGORITMO GENÉTICO (MLR-GA)

A Figura 28 aponta as curvas de treinamento considerando os valores de internações hospitalares observados e estimados, para 2 anos de dados coletados e simulados.

Figura 28 – Curvas de Treinamento entre as variáveis Simuladas e Observadas

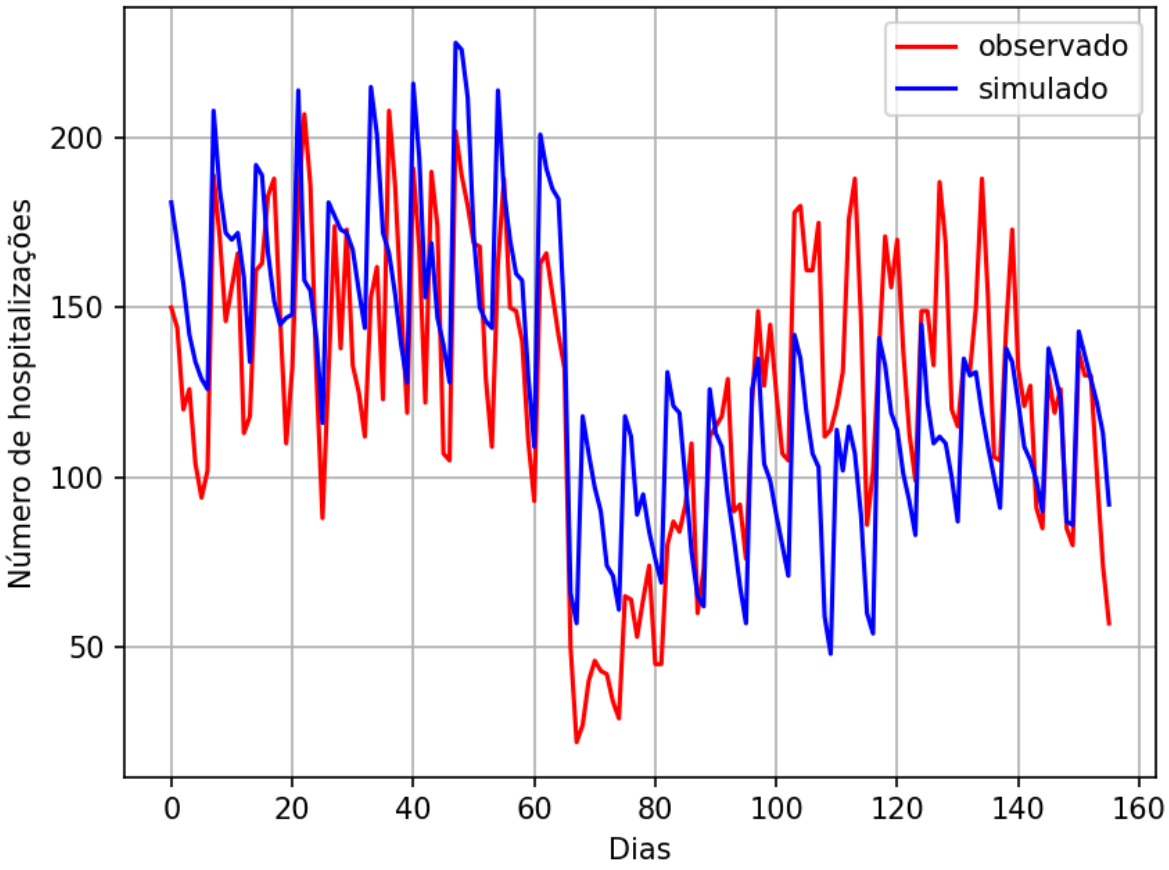


Fonte: Autoria própria.

Observa-se na Figura 28 que em torno do dia 200 o número de hospitalizações observado foi maior do que o simulado, em torno de 150 casos. Porém, entre o dia 200 e 300, os valores simulados foram superiores do que os registrados.

A Figura 29 apresenta as curvas de teste para o número de hospitalizações diário.

Figura 29 – Curvas de Teste entre as variáveis Simuladas e Observadas

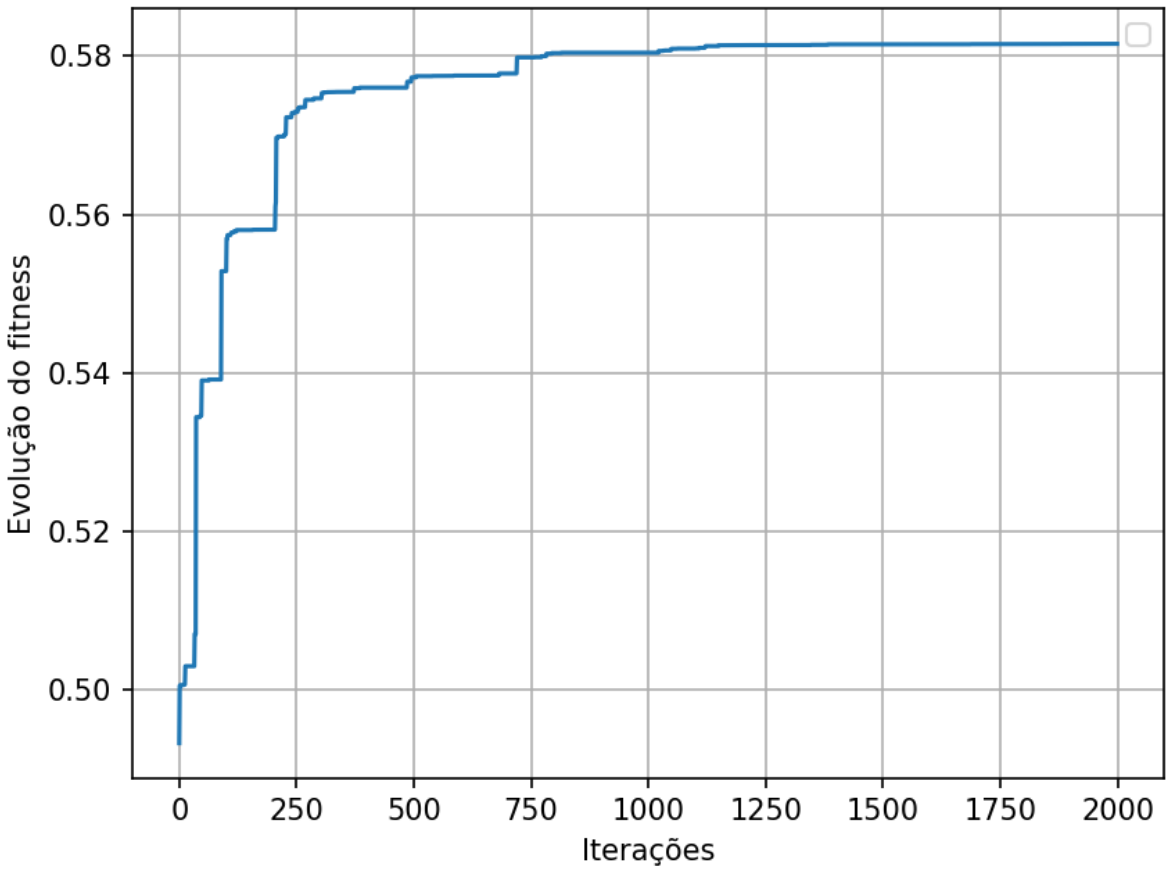


Fonte: Autoria própria.

Ao analisar a Figura 29 observa-se que os valores estimados pelo modelo foram maiores entre 40 e 60 dias. Após, entre os dias 60 e 80, há uma queda, apresentando os menores valores para a saída, de modo que os valores observados foram menores em relação aos simulados. Ainda, observa-se na figura que entre 100 e 140 dias, houve um período longo em que os valores apresentaram certa estabilidade, os observados com valores mais altos, enquanto os simulados, os valores mais baixos.

A Figura 30 apresenta os valores da evolução do *fitness* para o melhor indivíduo mediante o modelo MLR otimizado pelo GA.

Figura 30 – Evolução do Fitness para o Melhor Indivíduo

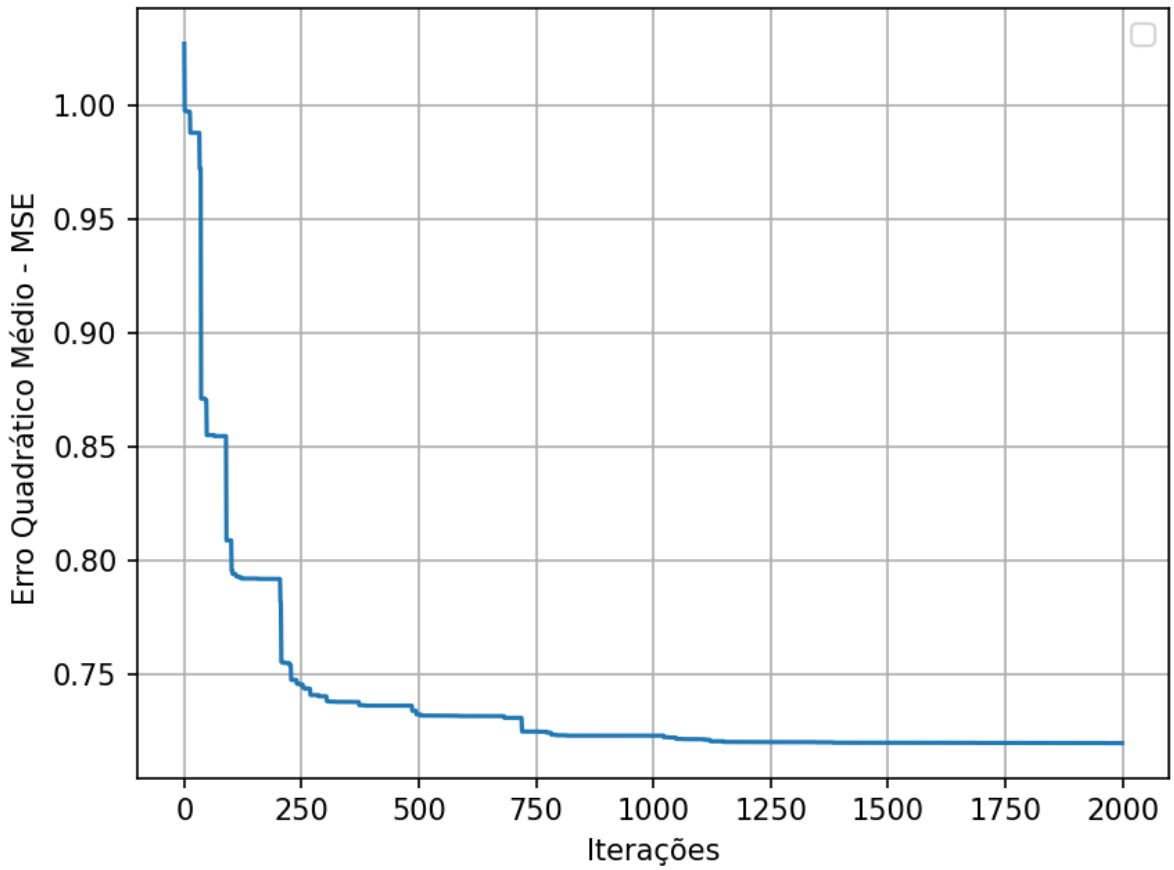


Fonte: Autoria própria.

Ao analisar a Figura 30 observa-se que a inicialização do algoritmo ocorreu com um *fitness* de aproximadamente 0,49, sendo que até atingir 1250 iterações, houve uma rápida evolução, quando o *fitness* estabilizou em pouco mais de 0,58. Ainda, observa-se que em 2000 interações, houve a convergência do algoritmo.

A Figura 31 apresenta a curva de evolução do MSE para o melhor indivíduo otimizado pelo GA.

Figura 31 – Evolução do Custo para o Melhor Indivíduo MLR-GA

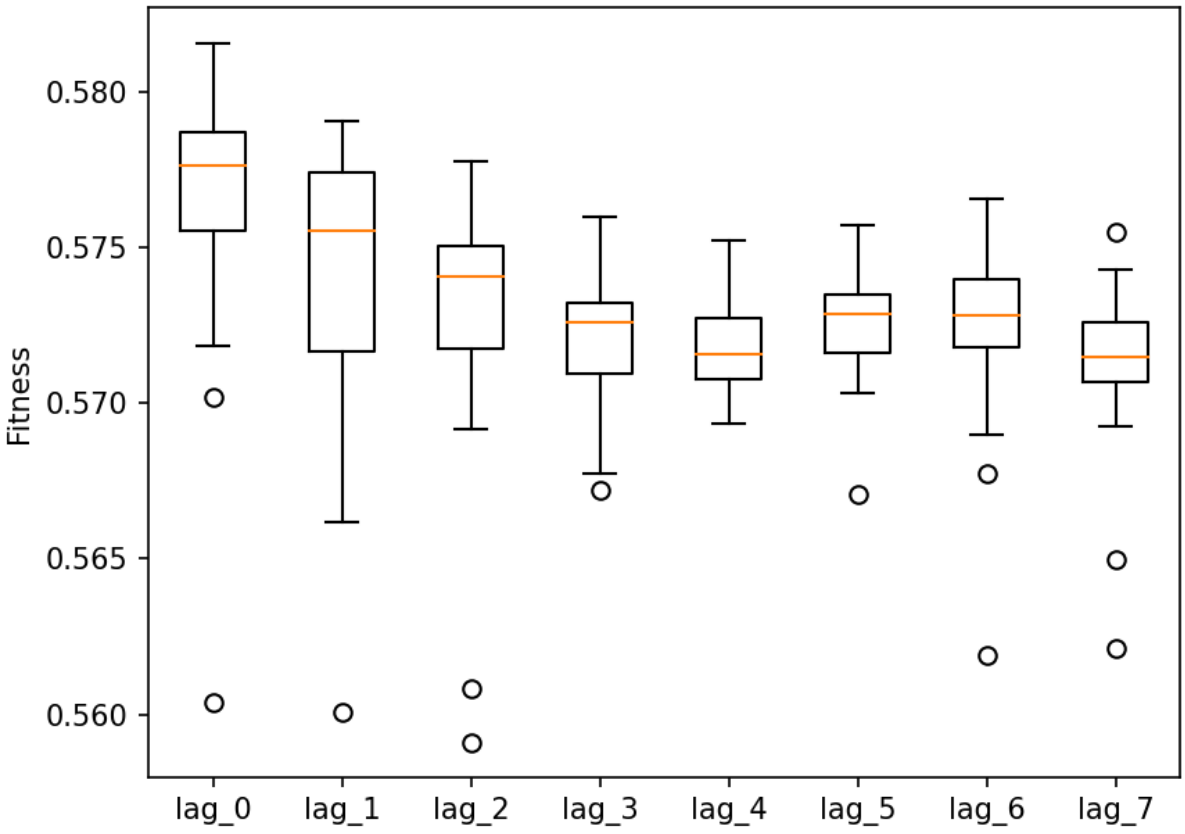


Fonte: Autoria própria.

Na Figura 31 a solução inicial apresentou um Erro Quadrático Médio (MSE) maior que 1. Esse erro é reduzido drasticamente entre a iteração 0 e 250, sendo que em aproximadamente 750 iterações, ele passa a ser menor que 0,65.

A Figura 32 apresenta a dispersão do *fitness* para os *lags*.

Figura 32 – Dispersão do Fitness para os *lags*

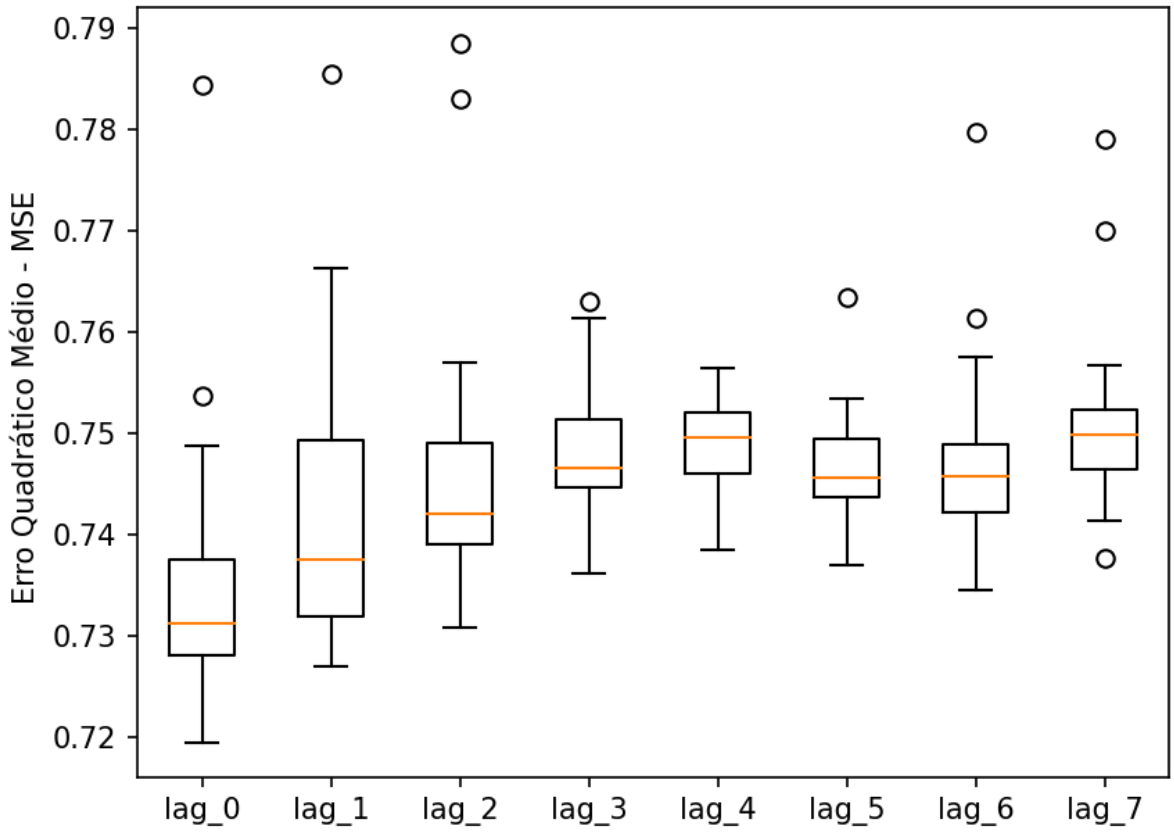


Fonte: Autoria própria.

É possível observar ao analisar a Figura 32, que para o *lag* 0, houve o melhor *fitness* médio e máximo (aproximadamente 0,583). De forma similar, para o *lag* 7 houve o pior desempenho geral, ainda que o *lag* 2 tenha 2 desempenhos finais relativamente ruins. Ainda, é possível observar que o grau de dispersão entre os valores encontrados foi baixo, pois há uma baixa oscilação entre as médias dos resultados obtidos. A máxima dispersão foi relativa ao *lag* 1.

Por fim, a Figura 33 apresenta o boxplot do MSE. Nota-se que para o *lag* 2 houve o pior erro máximo (aproximadamente 0,789), enquanto a pior média foi referente ao *lag* 7.

Figura 33 – Dispersão do Custo para 8 *lags*



Fonte: Autoria própria.

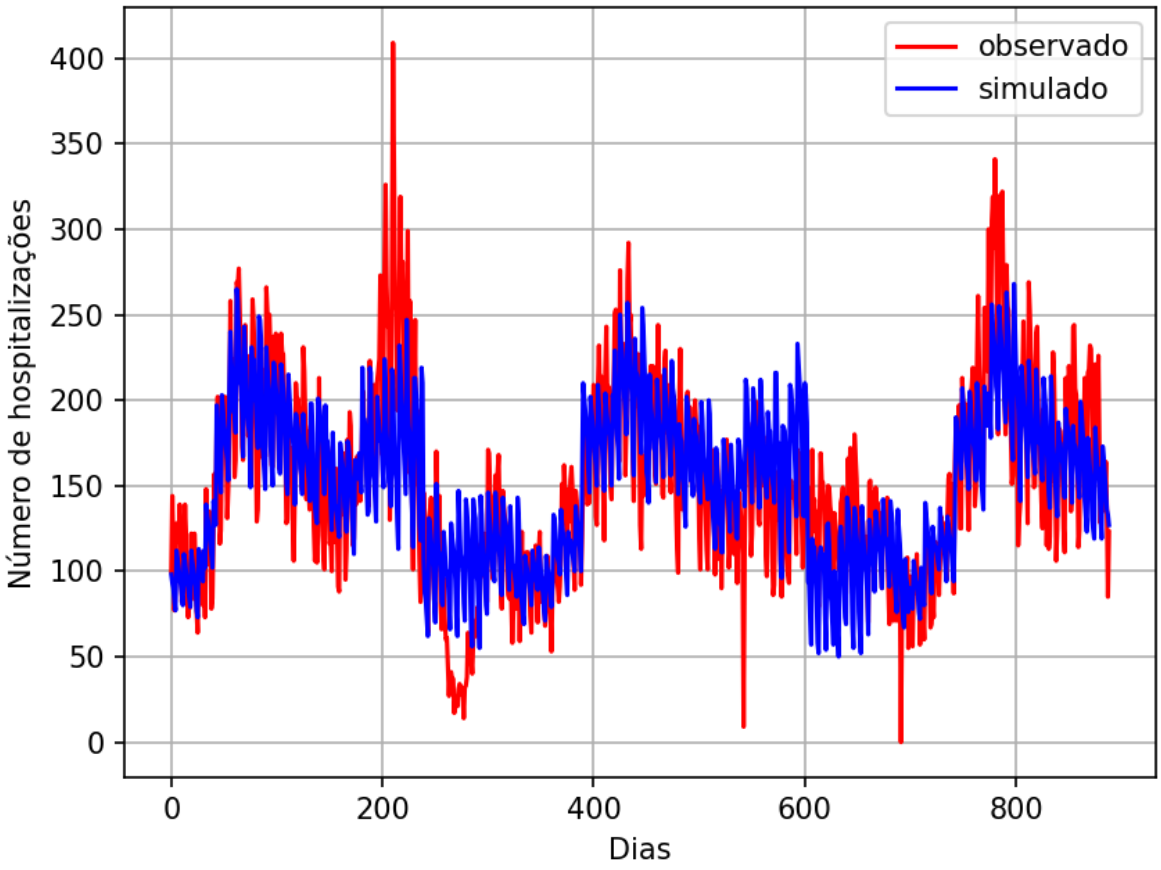
Ainda, a Figura 33 aponta que o menor erro foi para o *lag* 0, com o menor Erro Quadrático Médio (aproximadamente 0.72).

A seção a seguir, apresenta as figuras com os resultados do Regressão Linear Múltipla otimizada pelo Enxame de Partículas (MLR-PSO) para realizar a previsão de internações hospitalares relacionadas a doenças respiratórias.

4.3 REGRESSÃO LINEAR MÚLTIPLA OTIMIZADA PELO ENXAME DE PARTÍCULAS (MLR-PSO)

A Figura 34 aponta as curvas de treinamento entre os valores observados e simulados para 2 anos de dados.

Figura 34 – Curvas de Treinamento entre as variáveis Simuladas e Observadas

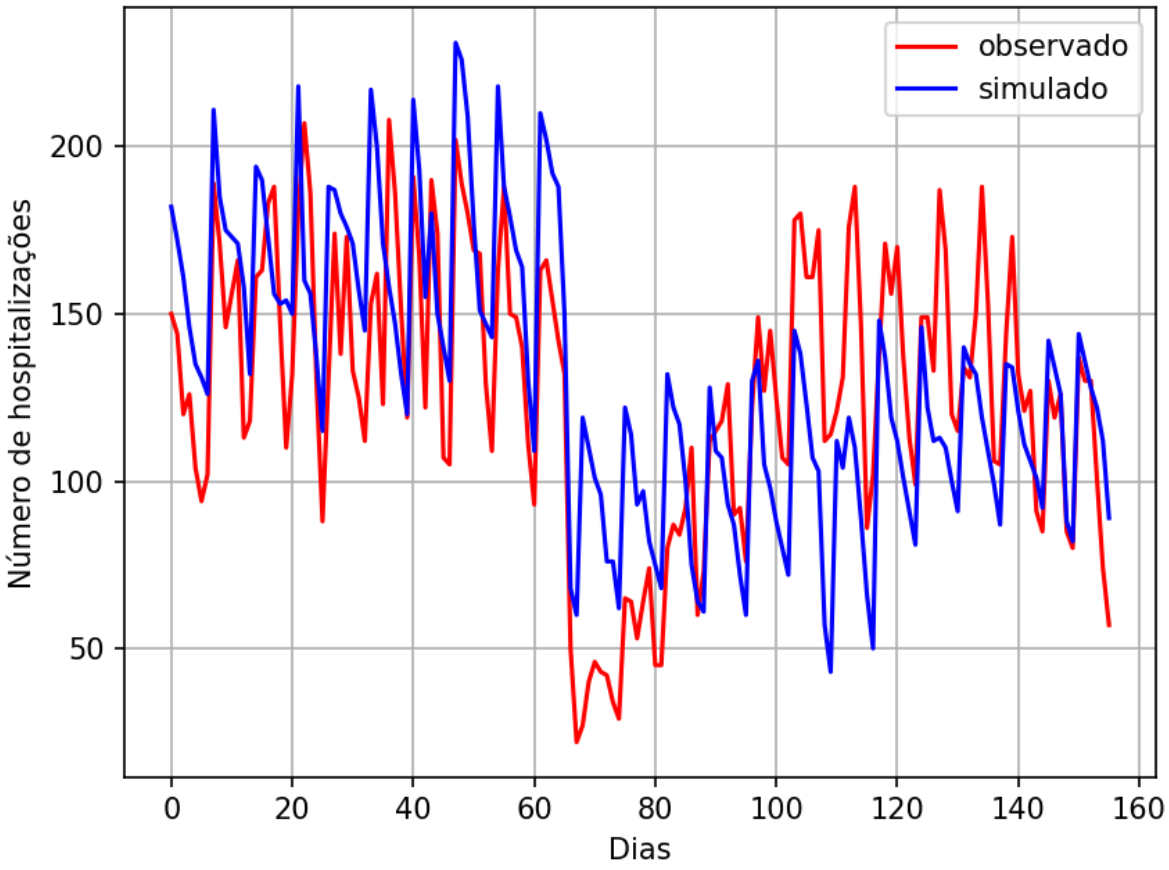


Fonte: Autoria própria.

Observa-se na Figura 34 que em torno do dia 200, o número de hospitalizações observadas foi maior do que os valores estimados, enquanto que entre os dias 200 e 300, os valores simulados foram superiores aos registrados.

A Figura 35 apresenta as curvas de teste.

Figura 35 – Curvas de Teste entre as variáveis Simuladas e Observadas

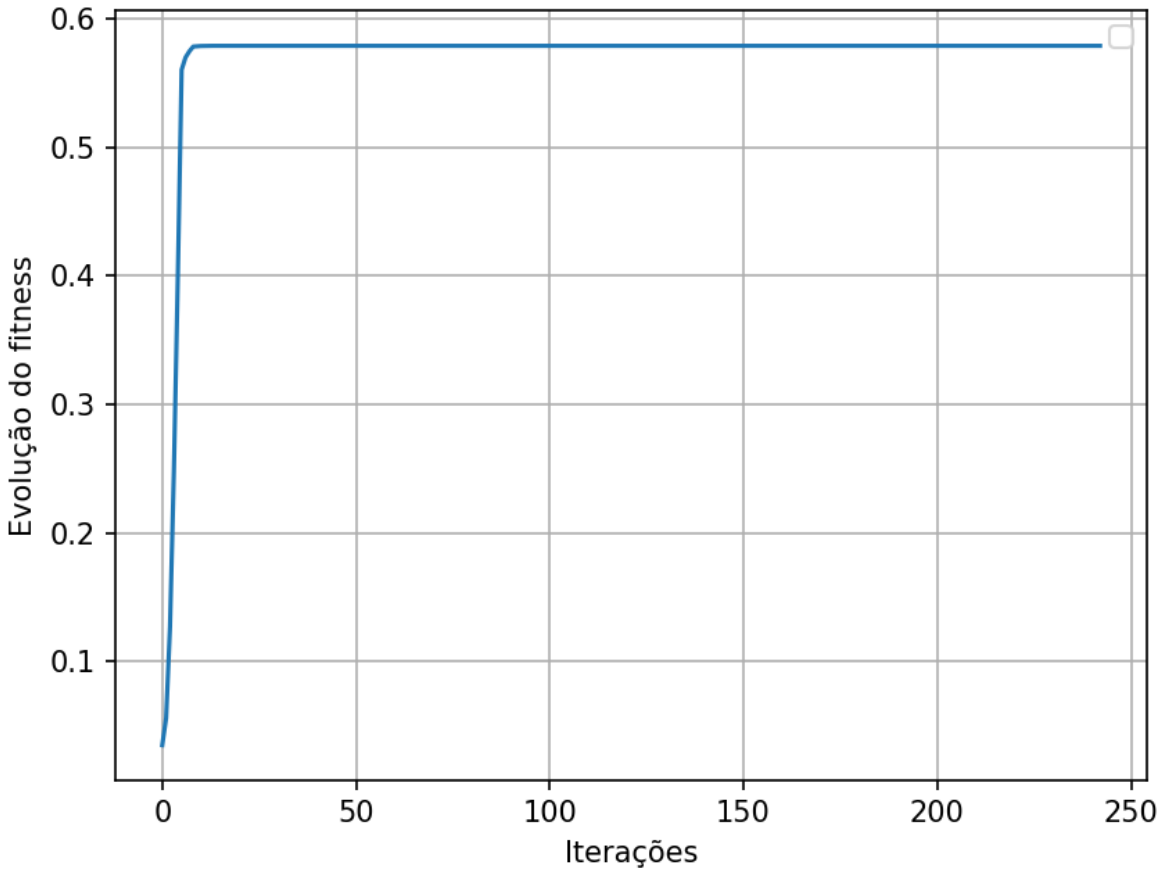


Fonte: Autoria própria.

Observa-se na Figura 35 que os os valores estimados foram maiores que os reais nos primeiros 60 dias. Após, entre os dias 60 e 80, houve uma queda, apresentando os menores valores para as saídas. Porém, os valores voltaram a crescer após 80 dias, sendo que entre os dias 100 e 120, os valores estimados foram menores quando comparados aos observados.

A Figura 36 apresenta os valores da evolução do *fitness* para o melhor agente mediante o modelo MLR otimizado pelo PSO.

Figura 36 – Evolução do Fitness para o Melhor Indivíduo

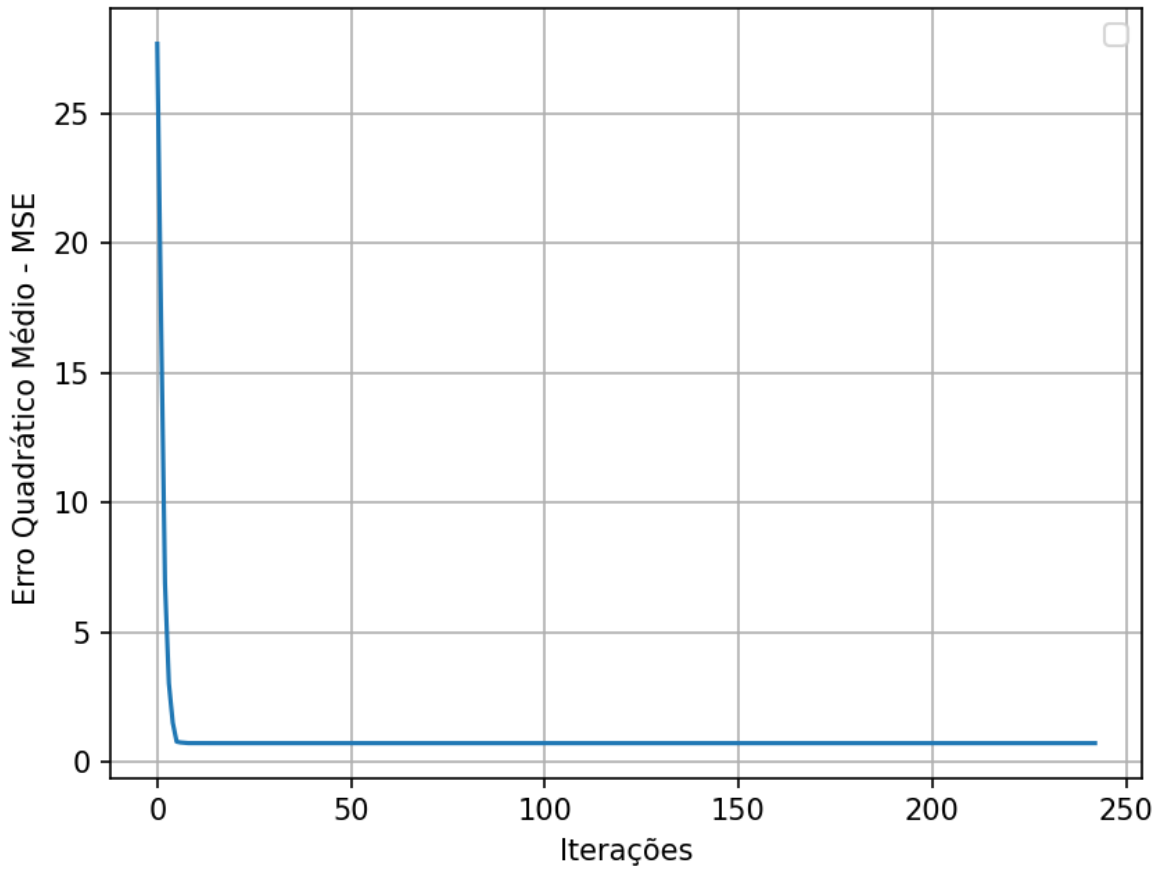


Fonte: Autoria própria.

A Figura 36 mostra que a inicialização do algoritmo ocorreu com um *fitness* de aproximadamente 0,04. Após, há uma evolução do *fitness*, chegando a aproximadamente 0,58, quando ocorre a iteração 20. Então, o *fitness* estabilizou em aproximadamente 0,58, até a iteração 240, em que houve a convergência do algoritmo. Note que o PSO convergiu muito mais rapidamente que as propostas evolutivas.

Observa-se na Figura 36 a curva de evolução do custo para o melhor indivíduo.

Figura 37 – Evolução do Custo para o Melhor Indivíduo

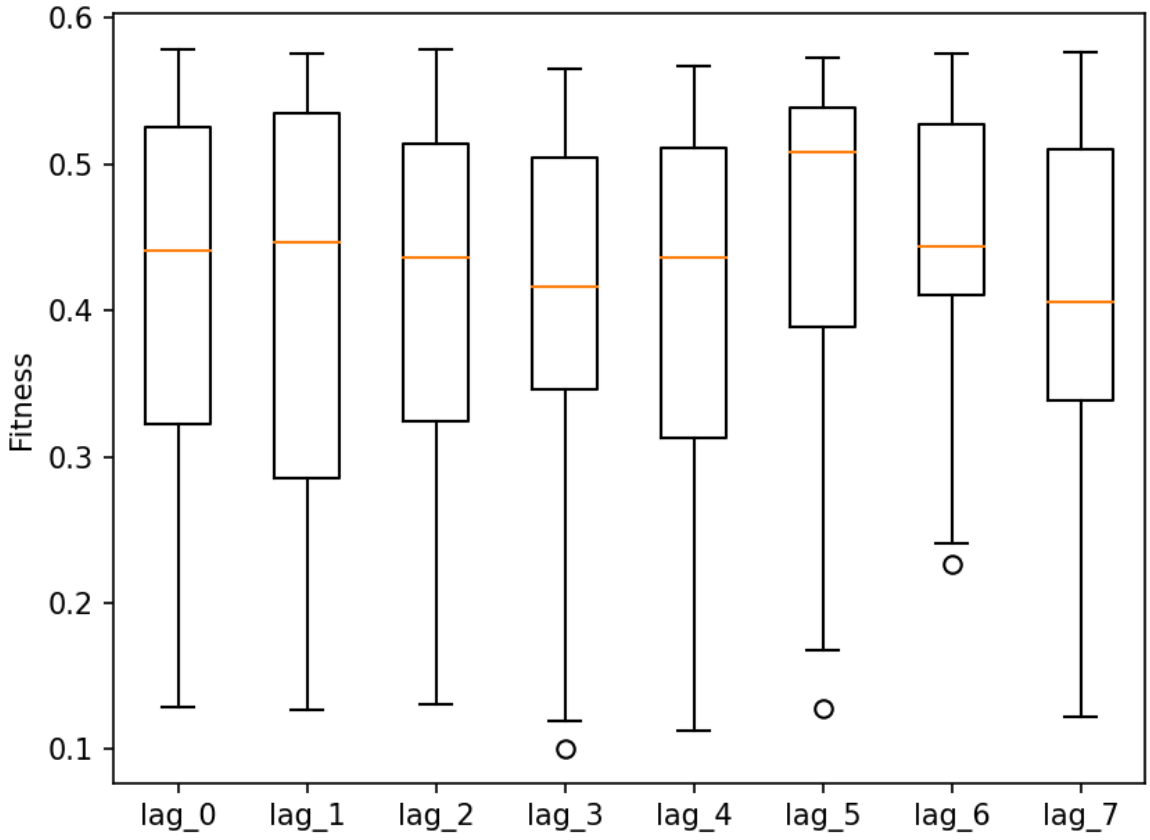


Fonte: Autoria própria.

A curva apresentada na Figura 36 aponta como solução inicial um MSE maior que 25. Esse erro diminui rapidamente em menos que 10 iterações, passando a menor que 1 por todas as iterações seguintes.

A Figura 38 apresenta o gráfico de dispersão do *fitness* para os 8 *lags* considerados.

Figura 38 – Dispersão do Fitness para os *lags*

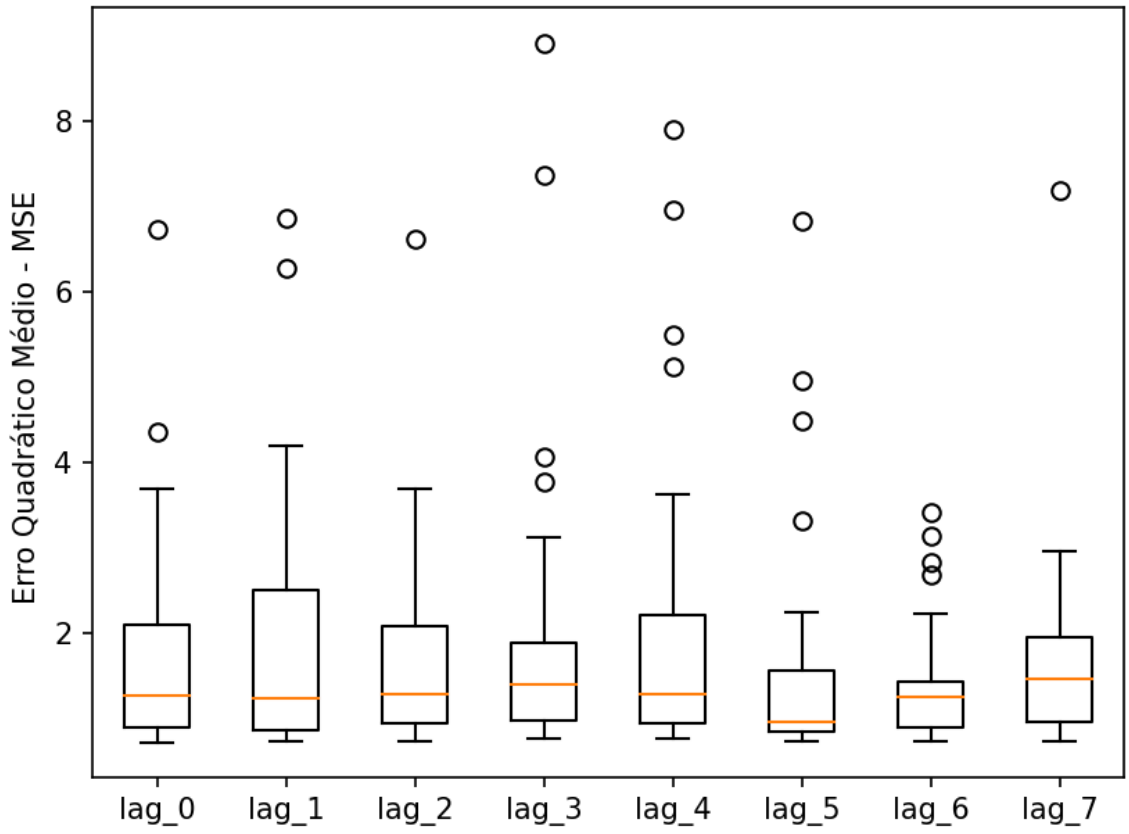


Fonte: Autoria própria.

É possível observar na Figura 38 que os resultados foram mais homogêneos com relação a todos *lags*, embora com uma dispersão muito grande.

Por fim, a Figura 38 apresenta o gráfico de dispersão do custo para 8 *lags*.

Figura 39 – Dispersão do Custo para 8 *lags*



Fonte: Autoria própria.

É possível observar que a média do erro foi similar para todos os *lags*, embora os valores alcançados estejam mais dispersos. Em termos de dispersão, *lag* 6 foi o mais uniforme.

4.4 COMPARATIVO DOS RESULTADOS

Esta sessão descreve os resultados tabulados a partir das oito métricas de avaliação presentes no *framework*, apresentadas na metodologia deste estudo. A Tabela 1 apresenta os resultados obtidos para o *lag* zero entre as 3 variações do modelo utilizadas: MLR-PSO, MLR-GA e MLR-DE. Para análise geral foi considerado o MSE como a métrica absoluta, pois é a mais utilizada na literatura.

Tabela 1 – Resultados obtidos para o *lag* 0

Métrica	PSO	GA	DE
AE	4518,00	4352,00	4335,00
MSE	1180,22	1109,99	**1109,78**
MAPE	125,82%	125,84%	125,84%
ARV	0,67	0,67	0,66
IA	0,81	0,82	0,82
MAE	28,96	27,90	27,79
RMSE	34,35	33,32	33,31
MEAN	131,00	129,53	129,20

Fonte: Autoria própria.

A métrica MSE, na Tabela 1, apontou o melhor resultado mediante uso do modelo MLR-DE. Este resultado é próximo ao observado no modelo MLR-GA, e apresenta uma melhora significativa em relação ao modelo MLR-PSO.

Como observado na Tabela 2, o MSE também apontou o modelo MLR-DE com o melhor resultado, para o *lag* 1.

Tabela 2 – Resultados obtidos para o *lag* 1

Métrica	PSO	GA	DE
AE	4465,00	4368,00	4337,00
MSE	1156,93	1145,05	**1120,13**
MAPE	126,05%	126,06%	126,06%
ARV	0,69	0,65	0,65
IA	0,81	0,82	0,82
MAE	28,62	28,00	27,80
RMSE	34,01	33,84	33,47
MEAN	130,71	130,16	129,72

Fonte: Autoria própria.

Ainda, também é possível notar na Tabela 2, que o resultado é 2,17% ao observado no modelo MLR-GA, e 3,18% melhor em relação ao modelo MLR-PSO.

A Tabela 3, também apresenta, no MSE, o modelo MLR-DE com o melhor resultado para o *lag* 2.

Tabela 3 – Resultados obtidos para o *lag* 2

Métrica	PSO	GA	DE
AE	4380,00	4257,00	4271,00
MSE	1135,97	1092,98	**1087,25**
MAPE	126,16%	126,16%	126,17%
ARV	0,66	0,67	0,66
IA	0,82	0,82	0,82
MAE	28,08	27,29	27,38
RMSE	33,70	33,06	32,97
MEAN	130,87	131,15	130,31

Fonte: Autoria própria.

Observa-se na Tabela 3, que o resultado obtido pele modelo MLR-DE é 0,52% melhor do que o observado no modelo MLR-GA, e 4,28% melhor em relação ao modelo MLR-PSO.

A Tabela 4 apresenta para o *lag* 3 o melhor resultado para o modelo MLR-PSO, conforme observado na métrica MSE.

Tabela 4 – Resultados obtidos para o *lag* 3

Métrica	PSO	GA	DE
AE	4071,00	4179,00	4293,00
MSE	**1018,80**	1062,06	1086,26
MAPE	126,33%	126,31%	126,32%
ARV	0,77	0,68	0,67
IA	0,82	0,82	0,82
MAE	26,10	26,79	27,52
RMSE	31,92	32,59	32,96
MEAN	128,76	130,91	130,22

Fonte: Autoria própria.

A Tabela 4, mostra que o resultado obtido pele modelo MLR-PSO é 4,17% melhor do que o observado no MLR-GA e 6,21% melhor em relação ao MLR-DE.

De modo similar, a Tabela 5 apresenta os erros para o *lag* 4, sendo o melhor resultado referente ao modelo MLR-GA.

Tabela 5 – Resultados obtidos para o *lag* 4

Métrica	PSO	GA	DE
AE	4360,00	4136,00	4301,00
MSE	1134,87	**1039,44**	1098,04
MAPE	126,62%	126,62%	126,63%
ARV	0,68	0,69	0,69
IA	0,82	0,83	0,82
MAE	27,95	26,51	27,57
RMSE	33,69	32,24	33,14
MEAN	131,88	130,90	130,69

Fonte: Autoria própria.

A Tabela 5, indica que o resultado obtido pelo MLR-GA é 5,3% superior ao do MLR-DE e 8,4% melhor em relação ao MLR-PSO.

Para o *lag* 5 a Tabela 6 também apresentou um MSE com melhor resultado para o modelo MLR-GA.

Tabela 6 – Resultados obtidos para o *lag* 5

Métrica	PSO	GA	DE
AE	4616,00	4089,00	4278,00
MSE	1214,37	**1005,46**	1081,59
MAPE	126,63%	126,66%	126,66%
ARV	0,71	0,71	0,68
IA	0,80	0,83	0,82
MAE	29,59	26,21	27,42
RMSE	34,85	31,71	32,89
MEAN	134,25	130,44	130,76

Fonte: Autoria própria.

Ainda, na Tabela 6, é possível observar que o resultado obtido pelo MLR-GA é superior em 7,03% em relação ao MLR-DE e 17,20% melhor em relação ao MLR-PSO.

A Tabela 7 apontou, para o *lags* 6 e 7, que o MLR-GA também obteve o melhor resultado.

Tabela 7 – Resultados obtidos para o *lag* 6

Métrica	PSO	GA	DE
AE	4128,00	4140,00	4288,00
MSE	1035,49	**1030,59**	1081,44
MAPE	126,32%	126,29%	126,30%
ARV	0,67	0,70	0,69
IA	0,83	0,82	0,82
MAE	26,46	26,54	27,49
RMSE	32,18	32,10	32,89
MEAN	128,53	130,91	130,35

Fonte: Autoria própria.

O melhor resultado mediante análise do Tabela 7 é 0,47% superior ao MLR-PSO e 4,70% melhor que o MLR-DE.

Tabela 8 – Resultados obtidos para o *lag* 7

Métrica	PSO	GA	DE
AE	4436,00	4156,00	4305,00
MSE	1153,69	**1052,60**	1097,56
MAPE	126,61%	126,61%	126,62%
ARV	0,68	0,70	0,68
IA	0,81	0,82	0,82
MAE	28,44	26,64	27,60
RMSE	33,97	32,44	33,13
MEAN	131,11	130,96	130,12

Fonte: Autoria própria.

A análise da Tabela 8 aponta o MLR-GA com um MSE de 4,09% menor em relação ao MLR-DE e 8,76% inferior em relação ao resultado encontrado pelo modelo MLR-PSO.

No contexto geral, MLR-GA foi o modelo que apresentou na maioria das simulações, para os *lags* 4, 5, 6 e 7. Em seguida, o MLR-DE apresentou o melhor resultado para três casos (*lags* 0, 1 e 2). Por fim, o MLR-PSO apresentou apenas o melhor resultado no log 3. Dentro desta perspectiva, o melhor resultado geral foi relativo ao *lag* 5, otimizado pelo GA. Em contrapartida, o maior erro foi obtido com o *lag* 1.

Ressalta-se que o modelo MLR-PSO, como observado nos resultados apontados por este estudo, apresentou alto grau de dispersão em relação aos erros ocorridos nas previsões. Ainda, o modelo também apresentou uma inicialização ruim para as soluções. Essas questões podem ter influenciado diretamente nos resultados de desempenho deste modelo.

A avaliação final aponta que para este estudo de casos os algoritmos evolutivos se comportaram melhor que o PSO, embora uma investigação paramétrica mais aprofundada seja necessária. O *framwork* apresenta diversas variações de todos os algoritmos de otimização mas apenas uma foi aplicada como forma de apresentar a originalidade da proposta e sua execução prática.

5 CONCLUSÕES E PERSPECTIVAS

Este trabalho teve como objetivo o desenvolvimento de um framework que possuísse uma interface instintiva e de fácil utilização para aplicação em estimação de valores futuros, como o de internações por doenças respiratórias. A metodologia empregada foi desenvolvida com base no modelo MLR, sendo utilizada diversas variantes do algoritmo genético (GA), evolução diferencial (DE) e da otimização por enxame de partículas (PSO) para a sua calibração.

Desenvolveu-se um programa com uma interface gráfica para configuração e customização dos algoritmos passíveis de serem utilizados, o qual apresenta uma etapa para tratamento e pré-processamento dos dados, de forma a obter dados numéricos e gráficos que facilitem a análise dos resultados.

Como meio de apresentar resultados práticos do modelo, foram realizados testes com os algoritmos utilizando uma base de internações por doenças respiratórias em São Paulo (saída ou alvo), de modo que as entradas são as variáveis climáticas e concentração de poluente (MP 10).

Considerando a métrica MSE, os algoritmos tiveram resultados interessantes, podendo variar a melhor escolha a depender do log analisado. De uma forma geral, os testes do modelo MLR-GA apresentaram os melhores resultados, sendo superior na maior parte dos casos. Porém, como os valores do MLR-DE e do MLR-PSO foram também adequados, pode-se sugerir que os mesmos possuem a eficácia necessária para realização de tarefas similares. Dessa maneira, o ideal em problemas correlatos é experimentar todas as possibilidades disponíveis no framework.

A seção a seguir apresenta as sugestões para trabalhos futuros.

5.1 TRABALHOS FUTUROS

Por meio do desenvolvimento deste trabalho de conclusão de curso, sugerem-se algumas oportunidades de novos desenvolvimentos para o framework.

- Adição de novos algoritmos de otimização;
- Acréscimo de mais abordagens como o modelo linear generalizado (MLG);
- Adição de novas entradas que tenham relação com a previsão que se deseja realizar;

- Novos tratamentos nos dados de entrada, como a binarização dos dias da semana;
- Acréscimo de testes estatísticos, como o teste de Friedman.

Essas oportunidades corroborarão para o aperfeiçoamento do framework apresentado.

REFERÊNCIAS

BELOTTI, J. T. et al. Air pollution epidemiology: A simplified Generalized Linear Model approach optimized by bio-inspired metaheuristics. **Environmental Research**, v. 191, p. 110106, 2020. DOI: https://doi.org/10.1016/j.envres.2020.110106.

CASTANHO, D. S. **Previsão do estado de carga de bateria aplicada a veículos elétricos usando modelos lineares auto-ajustados através de algoritmos de otimização**. 2019. F. 77. Diss. (Mestrado) – Universidade Tecnológica Federal do Paraná, Ponta Grossa.

CASTANHO, D. S. et al. Aplicação de um controlador pid fuzzy adaptativo otimizado via algoritmo genético à um motor de corrente contínua didático. **XIII SIMMEC 2018**, p. 1–15, 2018.

CETESB, C. d. T. d. S. A. **Qualidade do ar no estado de São Paulo**. [S.l.], 2016.

CLERC, M.; KENNEDY, J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. **IEEE transactions on Evolutionary Computation**, v. 6, p. 58–73, 2002.

COELHO, I. et al. OptFrame: A computational framework for combinatorial optimization problems. **Proceedings of the 7th ALIO/EURO Workshop**, p. 57–59, 2011.

DATASUS, D. d. I. d. S. ú. d. S. **Informações de Saúde**. [S.l.], 2016.

DE CASTRO, L. N. **Fundamentals of natural computing**: basic concepts, algorithms, and applications. [S.l.]: Chapman e Hall/CRC, 2006. P. 696.

HAMAMOTO, A. H. **Aplicação de Algoritmos Genéticos para Auxiliar na Gerência de Redes**. 2014. F. 67. Diss. (Mestrado) – TCC (Bacharelado) - Curso de Ciência da Computação, Universidade Estadual de Londrina, Ponta Grossa.

HASSAN, M. et al. Optimal Design of Electric Power Assisted Steering System (EPAS) Using GA-PID Method. **Procedia Engineering**, v. 41, p. 614–621, 2012.

HOLLAND, J. H. Genetic Algorithms. **Scientific American**, v. 267, p. 66–73, 1992.

HU, Y. et al. A technique for dynamic battery model identification in automotive applications using linear parameter varying structures. **Control Engineering Practice**, v. 17, n. 10, p. 1190–1201, 2009.

J.H. AL GIZI, A. et al. Integrated PLC-fuzzy PID Simulink implemented AVR system. **International Journal of Electrical Power & Energy Systems**, v. 69, p. 313–326, 2015.

KENNEDY, J. Particle swarm optimization. In: SAMMUT, C.; WEBB, G. I. (Ed.). **Encyclopedia of Machine Learning**. [S.l.]: Springer US, 2010. P. 760–766.

LAZZARIN, L. A. **Método ensemble baseado em redes neurais artificiais para estimação de internações por doenças respiratórias**. 2019. Diss. (Mestrado) – Universidade Tecnológica Federal do Paraná, Ponta Grossa.

MARTINS, M. et al. PSO with path relinking for resource allocation using simulation optimization. **Computers & Industrial Engineering**, v. 65, n. 2, p. 322–330, 2013.

MICHALEWICZ, Z. Evolution Strategies and Other Methods. In: MICHALEWICZ, Z. (Ed.). **Genetic Algorithms + Data Structures = Evolution Programs**. [S.l.]: Springer Berlin Heidelberg, 1996. P. 159–177.

MORETTIN, P. A.; TOLOI, C. **Análise de séries temporais**. [S.l.]: ABE- Projeto Fisher e Editora Edgard Blucher, 2006.

NEUHAUS, J.; MCCULLOCH, C. Generalized linear models. **Wiley Interdisciplinary Reviews: Computational Statistics**, v. 3, n. 5, p. 407–413, 2011.

ÖZTÜRK, N.; ÇELIK, E. Speed control of permanent magnet synchronous motors using fuzzy controller based on genetic algorithms. **International Journal of Electrical Power & Energy Systems**, v. 43, p. 889–898, 2012.

PYQT. PyQt: Reference Guide, 2020. Disponível em: https://www.riverbankcomputing.com/static/Docs/PyQt5/.

RAHMAN, M. A.; ANWAR, S.; IZADIAN, A. Electrochemical model parameter identification of a lithium-ion battery using particle swarm optimization method. **Journal of Power Sources**, v. 307, p. 86–97, 2016.

SILVA, M. A. L. et al. Hybrid metaheuristics and multi-agent systems for solving optimization problems: A review of frameworks and a comparative analysis. **Applied Soft Computing**, v. 71, p. 433–459, 2018.

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

UYANIK, G. K.; GÜLER, N. A study on multiple linear regression analysis. **Procedia-Social and Behavioral Sciences**, v. 106, p. 234–240, 2013.

WAGNER, S.; AFFENZELLER, M. Heuristiclab Grid - A Flexible And Extensible Environment For Parallel Heuristic Optimization. **Systems Science**, v. 30, 2004.

WAGNER, S.; BEHAM, A. et al. HeuristicLab 3.3: A unified approach to metaheuristic optimization. **Actas del séptimo congreso español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados**, 2010.

ÍNDICE REMISSIVO

AE, 27, 37, 40, 43, 45
AI, 14
ARV, 28, 45

DE, 13, 14, 20, 29, 32, 33, 48

EA, 16

GA, 13, 14, 16, 17, 19, 20, 29, 32, 33, 55, 56
GUI, 31

MAE, 28, 45
MAPE, 28, 45
MLR, 13, 14, 16, 24, 26, 29, 48, 55, 70
MSE, 27–29, 37, 40, 43, 45, 49–51, 66, 67, 70

PSO, 13, 14, 21, 22, 29, 32, 33, 42

RMSE, 28, 45

UTFPR, i