Relatório de Atividade
Sintonia de Ganhos para Leis de Controle Não-Lineares em Robôs Móveis de Tração Diferencial

Matheus Lucas Tavares de Farias
Prof. AMN Lima
Automação Inteligente
11 de Agosto de 2025

***Resumo*—Este relatório apresenta a formulação, implementação e análise do problema de ajuste de ganhos de leis de controle não-lineares aplicadas a robôs móveis com tração diferencial. O estudo é baseado em três abordagens extraídas da literatura. Em todos os casos, o problema de sintonia dos ganhos foi resolvido considerando restrições físicas nas velocidades linear e angular, garantindo a validade do modelo cinemático do robô. A análise comparativa foi realizada entre técnicas de otimização tradicionais (*fmincon*) e heurísticas (*Speedyga*), utilizando a linguagem de programação Python. Os experimentos foram conduzidos em ambiente simulado com o modelo cinemático *unicycle*, e os resultados indicam os impactos práticos da escolha dos ganhos e do método de otimização na performance dos controladores.**

***Index Terms*—Sintonia de ganhos, otimização, algorítimo genético, modelo *unicycle*.**

I. Introdução

A navegação autônoma de robôs móveis permanece como um dos principais focos de pesquisa em robótica, devido à sua ampla aplicabilidade em áreas como logística, vigilância, agricultura de precisão, robótica de serviços e exploração ambiental. Um desafio recorrente nesse campo é o controle de robôs com restrições não-holonômicas, especialmente em tarefas de rastreamento de alvos (go-to-goal).

A eficácia de um sistema de controle depende fortemente da escolha adequada de seus parâmetros ou ganhos, que influenciam diretamente o comportamento do robô em termos de estabilidade, desempenho e resposta a perturbações. No entanto, a sintonia desses ganhos não é trivial e exige abordagens sistemáticas que considerem tanto critérios de desempenho quanto limitações físicas do sistema, como velocidades linear e angular máximas admissíveis.

Este relatório tem como objetivo formalizar e resolver o problema de ajuste de ganhos para três leis de controle não lineares amplamente discutidas na literatura, propostas por [1], [2] e [3]. As duas primeiras já foram exploradas em trabalhos anteriores no contexto desta disciplina, enquanto a terceira se baseia em uma adaptação da abordagem de [1].

Em todos os casos, a sintonia dos ganhos foi tratada como um problema de otimização sujeito a restrições cinemáticas. A solução foi obtida por meio da comparação entre métodos tradicionais, como a otimização não linear com restrições via *fmincon*, e abordagens heurísticas baseadas em algoritmos genéticos, como o *SpeedyGA*. Todos os algoritmos foram implementados em Python, e os testes foram conduzidos em ambiente simulado com base no modelo cinemático *unicycle*, utilizando o robô Pioneer 3-DX como referência para definição das restrições.

II. Métodos de Otimização

Otimização é o processo de encontrar a melhor solução para um problema, considerando restrições e critérios definidos. Em geral, envolve ajustar variáveis de decisão para minimizar ou maximizar uma função objetivo, que representa custo, erro ou desempenho do sistema.

Esses métodos podem ser aplicados na sintonia de controladores, evitando a abordagem de tentativa e erro ao buscar soluções de forma sistemática e eficiente. Neste trabalho, eles foram utilizados para ajustar os parâmetros das leis de controle.

Os métodos utilizados foram: um tradicional baseado em gradiente, implementado com `scipy.optimize.minimize`, e um heurístico inspirado em algoritmos genéticos, adaptado da implementação *SpeedyGA*. Ambos visam identificar os conjuntos de ganhos que proporcionem melhor desempenho no rastreamento de alvos, respeitando as restrições do modelo cinemático do robô.

A. Otimização com Restrições - *fmincon*

Neste trabalho, a abordagem baseada no método *fmincon*, originario do MATLAB, foi implementada utilizando a função `minimize` da biblioteca `scipy.optimize` da linguagem Python, com o método *Sequential Least Squares Programming* (SLSQP). Essa abordagem permite resolver problemas de otimização contínua com restrições, sendo amplamente utilizada em aplicações de engenharia e controle.

A ideia central do método é ajustar automaticamente os ganhos dos controladores de modo a minimizar uma função

custo definida pelo projetista. Essa função custo reflete o desempenho do robô durante a tarefa de rastreamento de alvos e pode levar em conta critérios como o erro de posição ao longo do tempo, o tempo total de chegada ao destino, ou uma combinação ponderada desses fatores.

Além da função objetivo, é possível definir restrições que garantam que os parâmetros obtidos estejam dentro dos limites físicos e operacionais do sistema ou ainda restrições para garantir estabilidade e segurança.

Para utilizar o método, foram definidos:

- A função objetivo a ser minimizada;
- Os limites superiores e inferiores para cada ganho;
- Uma estimativa inicial dos ganhos.

A principal vantagem dessa abordagem é sua simplicidade de uso e sua boa performance em problemas com espaço de busca bem comportado. No entanto, por ser um método baseado em gradiente, ele pode ficar preso em mínimos locais, especialmente se a função custo for não convexa ou apresentar múltiplas soluções próximas.

A implementação foi realizada em Python, utilizando a função `scipy.optimize.minimize`, conforme documentado em [4].

B. Algoritmo Genético - SpeedyGA

Algoritmos genéticos são métodos heurísticos inspirados na seleção natural e na evolução biológica. Eles operam sobre uma população de soluções candidatas (indivíduos), que evoluem ao longo de várias gerações com o objetivo de encontrar uma solução que maximize uma determinada função objetivo.

O funcionamento desse tipo de algoritmo segue os seguintes passos principais:

- **Inicialização**: Uma população inicial é gerada aleatoriamente.
- **Avaliação (*Fitness*)**: Cada indivíduo da população é avaliado por meio da função de ajuste, que reflete o quão adequado aquele individuo está. Indivíduos com maior ajuste são considerados mais aptos.
- **Seleção**: Indivíduos com melhor desempenho têm maior probabilidade de serem selecionados para reprodução.
- **Cruzamento (*Crossover*)**: A partir dos indivíduos selecionados, são gerados novos indivíduos (filhos) combinando os parâmetros dos pais.
- **Mutação**: Após o cruzamento, os filhos passam por uma etapa de mutação, em que um ou mais parâmetros são alterados aleatoriamente com uma pequena probabilidade. Esse passo é importante para garantir a diversidade na população e evitar que o algoritmo convirja prematuramente para uma solução subótima.
- **Substituição**: A nova geração substitui a anterior, e o processo se repete até que um critério de parada seja atingido.

No escopo desse trabalho, o algorítimo genético utilizado foi o *SpeedyGA*, adaptado de uma implementação original em MATLAB proposta por Burjorjee [5]. Esse método foi reimplementado em Python para atender às necessidades deste projeto. No caso do *SpeedyGA*, cada indivíduo representa uma sequência binária.

A principal vantagem do *SpeedyGA* em comparação com métodos tradicionais baseados em gradiente é sua capacidade de realizar uma busca global, sendo menos sensível a mínimos locais. Além disso, ele não requer informações sobre derivadas da função objetivo, o que o torna adequado para problemas com funções custo não suaves ou ruidosas. Por outro lado, algoritmos genéticos tendem a demandar maior tempo computacional, especialmente quando aplicados a simulações complexas.

III. Modelo Cinemático e Leis de Controle

Neste trabalho, considera-se o modelo cinemático *unicycle* para representar o comportamento de um robô móvel com tração diferencial. Esse modelo é amplamente utilizado na literatura por capturar as principais características de mobilidade de robôs com restrições não holonômicas.

O modelo é descrito pelas equações 1, que relacionam as variáveis de estado à entrada de controle:

$$\begin{cases} \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \\ \dot{\theta} = \omega \end{cases} \tag{1}$$

em que (x, y) representa a posição do robô no plano, θ é a orientação (ângulo em relação ao eixo x), v é a velocidade linear, e ω é a velocidade angular.

Nesse sistema, verifica-se que as entradas de controle são as velocidades linear e angular. Para a realização do controle *go-to-goal*, os autores propõem leis de controle para essas grandezas.

Em todas as abordagens consideradas neste trabalho, o problema de controle é formulado com o auxílio de variáveis de erro no referencial do robô. Considerando que a posição do alvo no plano é dada por (x_g, y_g), definem-se as variáveis de erro apresentadas nas equações 2:

$$\begin{cases} e = \sqrt{(x_g - x)^2 + (y_g - y)^2}, \\ \alpha = \arctan\left(\dfrac{y_g - y}{x_g - x}\right) - \theta \end{cases} \tag{2}$$

onde e e α representam, respectivamente, o erro de posição e o erro de orientação do robô em relação ao alvo.

Com base nessas variáveis, três leis de controle não lineares foram implementadas. A primeira delas, apresentada por Aicardi *et al.* em [1], é expressa na equação 3:

$$\begin{cases} v = \gamma e\cos\alpha \\ \omega = k\alpha + \dfrac{v}{e\alpha}\sin\alpha\left(\alpha + h\left(\alpha + \theta\right)\right) \end{cases} \tag{3}$$

Para essa lei de controle, os parâmetros a serem ajustados são: γ, k e h.

A segunda lei de controle considerada é a proposta por Benbouabdallah *et al.* em [2], cuja formulação é apresentada na equação 4:

$$\begin{cases} v = K_v e \cos\alpha \\ \omega = K_\omega \alpha + \dfrac{v}{e}\sin\alpha \end{cases} \tag{4}$$

em que os parâmetros de controle são K_v e K_ω.

Por fim, também foi considerada a lei de controle derivada da proposta de Aicardi *et al.*, apresentada por Breno *et al.* em [3], cuja equação é mostrada em 5:

$$\begin{cases} v = \tau e \cos\alpha \\ \omega = \kappa \alpha + \dfrac{v}{e}\sin\alpha \end{cases} \tag{5}$$

em que os parâmetros de controle são τ e κ.

IV. Resultados

A execução dos métodos de otimização foi realizada utilizando a linguagem de programação Python, com o objetivo de obter os melhores parâmetros, dentro das limitações de cada algoritmo, para os ganhos das leis de controle apresentadas anteriormente.

Para sua aplicação, foi necessário definir uma cena de teste, estabelecer as funções objetivo para cada abordagem e determinar as restrições associadas a cada ganho.

Além disso, buscou-se garantir a integridade do modelo cinemático ao longo de toda a simulação. Para isso, adotaram-se como referência as limitações do robô Pioneer 3-DX, que, segundo sua documentação, possui restrições quanto às velocidades máximas linear e angular, sendo $v_{max} = 1{,}2\,\text{m/s}$ e $\omega_{max} = 300°/\text{s}$, respectivamente.

Com o intuito de reproduzir os resultados das referências, tentou-se utilizar as mesmas cenas de teste nelas presentes. Entretanto, dentro do escopo deste trabalho, apenas a referência [3] apresentou uma cena coincidente. A Figura 1 ilustra a cena de teste selecionada, utilizada em todos os casos analisados.

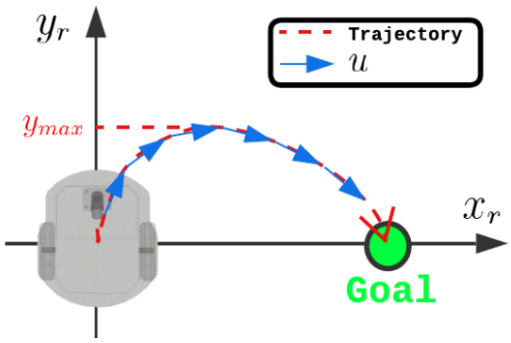

Figura 1. Cena de teste

Nessa cena, a posição inicial do robô é $(0, 0)$, com orientação igual a $\pi/2$, enquanto o alvo está localizado na posição $(1, 0)$.

Com o intuito de uniformizar todas as abordagens, os parâmetros de controle otimizados foram considerados como valores reais pertencentes ao intervalo $[0, 4]$. Este intervalo foi definido pelo autor deste relatório por ser suficientemente abrangente para representar os ganhos e coincidir com os valores adotados nas referências consultadas.

Um aspecto importante a ser mencionado é que o método de otimização baseado no algoritmo genético *SpeedyGA* representa cada indivíduo da população como uma sequência binária. Dessa forma, para obter os valores reais dos ganhos (x_{real}), realizou-se a conversão da representação inteira sem sinal $(x_{inteiro})$ correspondente à sequência binária. Essa conversão é descrita pela equação 6.

$$x_{real} = \frac{4}{2^{n_{bits}} - 1} \cdot x_{inteiro} \tag{6}$$

Nas execuções realizadas neste trabalho, cada ganho possuía uma representação binária de 20 bits, como sugerido em [3].

Por fim, as funções objetivo foram definidas com base nas referências, buscando respeitar as restrições impostas pelo modelo. Como resultado, foi elaborada uma função objetivo específica para cada abordagem adotada.

Vale destacar, ainda, que as funções de avaliação diferem entre os métodos. No algoritmo genético, utiliza-se uma função de ajuste, cuja performance melhora com o aumento do valor; já nos métodos tradicionais de otimização, emprega-se uma função de custo, na qual menores valores indicam melhores desempenhos. Assim, em geral, para cada abordagem, a função de custo pode ser considerada como o oposto da função de ajuste.

A. Resultados - Abordagem de Aicardi et al.

No trabalho de [1], não é apresentado nenhum método de otimização para a determinação dos ganhos da lei de controle. Por esse motivo, as funções-objetivo foram definidas pelo autor deste relatório, de forma a garantir que não houvesse violação das restrições de velocidade.

Para isso, a função de ajuste (F) foi definida conforme a Equação 7.

$$F = \frac{1}{|\max(v) - v_{ref}|} + \frac{1}{|\max(\omega) - \omega_{ref}|} \tag{7}$$

Em que $\max(v)$ e $\max(\omega)$ representam, respectivamente, os valores máximos das velocidades linear e angular ao longo de toda a simulação, enquanto v_{ref} e ω_{ref} são as velocidades máximas de referência para o robô Pioneer 3-DX.

Essa função de ajuste foi concebida de modo que os valores máximos das velocidades se aproximem o máximo possível de suas referências. Quando isso ocorre, a função atinge seu valor máximo, que corresponde ao objetivo final do algoritmo genético.

A função de custo (J), conforme mencionado anteriormente, foi definida como o oposto da função de ajuste, conforme a Equação 8.

$$J = -F \tag{8}$$

O objetivo, portanto, permanece o mesmo — encontrar parâmetros que mantenham as velocidades próximas aos valores de referência. No entanto, nesse caso, o algoritmo *fmincon* busca o mínimo da função.

Com as funções definidas, os algoritmos foram executados. Para o algoritmo genético, adotou-se uma população de 200 indivíduos, distribuídos em 40 gerações, com probabilidade de mutação de 3% e probabilidade de *crossover* de 100%.

Na Tabela I são apresentados os ganhos obtidos para as leis de controle em ambos os métodos.

Tabela I
GANHOS OBTIDOS PARA A ABORDAGEM DE AICARDI *et al.*

Método	γ	k	h
SpeedyGA	1.63	3.33	3.47
fmincon	3.38	2.74	2.26

Com esses parâmetros, foram calculados os valores máximos das velocidades para verificar a ocorrência de violações das restrições. Os resultados estão apresentados na Tabela II.

Tabela II
VALORES MÁXIMOS DE VELOCIDADE PARA A ABORDAGEM DE AICARDI *et al.*

Método	$\max(v)$ [m/s]	$\max(\omega)$ [°/s]
SpeedyGA	1.20	300.00
fmincon	2.17	300.00

Para avaliação visual, foi traçado o gráfico das trajetórias obtidas com ambos os métodos, apresentado na Figura 2.

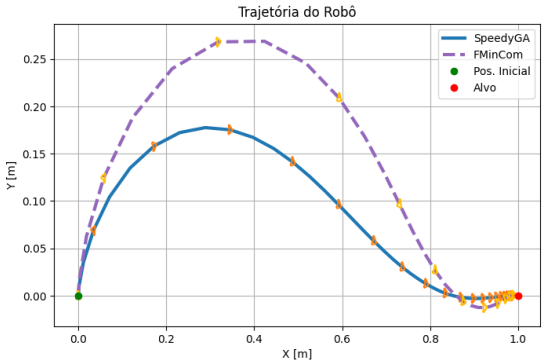

Figura 2. Trajetórias para a abordagem de Aicardi *et al.*

No gráfico, cada triângulo representa um ponto registrado em intervalos iguais de tempo.

B. Resultados - Abordagem de Benbouabdallah et al.

Em [2], os autores propõem uma função de custo para o processo de otimização, expressa pela Equação 9.

$$J = \frac{1}{2}\int \left(\frac{e}{e_o}\right)^2 + \left(\frac{\alpha}{\alpha_o}\right)^2 \tag{9}$$

Em que e_o e α_o correspondem, respectivamente, aos erros iniciais de posição e orientação.

O objetivo dessa função é minimizar os erros de posição e orientação no menor tempo possível. Entretanto, ela não considera as restrições físicas do modelo. Por esse motivo, o autor deste relatório optou por acrescentar dois termos adicionais à função de custo, induzindo os métodos de otimização a levar em conta também as velocidades linear e angular.

Assim, a nova função de custo é dada pela Equação 10.

$$J = \frac{1}{2}\int \left(\frac{e}{e_o}\right)^2 + \left(\frac{\alpha}{\alpha_o}\right)^2 - \frac{1}{|\max(v) - v_{ref}|} - \frac{1}{|\max(\omega) - \omega_{ref}|} \tag{10}$$

Como mencionado anteriormente, a função de ajuste (F) associada é simplesmente o oposto da função de custo, conforme a Equação 11.

$$F = -J \tag{11}$$

Com as funções definidas, procedeu-se à execução dos algoritmos de otimização. Para o algoritmo genético, adotou-se uma população de 100 indivíduos, distribuídos em 100 gerações, com probabilidade de mutação de 3% e probabilidade de *crossover* de 100%.

A Tabela III apresenta os ganhos obtidos para cada método.

Tabela III
GANHOS OBTIDOS PARA A ABORDAGEM DE BENBOUABDALLAH *et al.*

Método	K_v	K_ω
SpeedyGA	1.70	3.33
fmincon	1.87	2.21

Simulando o modelo com esses parâmetros, obtiveram-se os valores máximos das velocidades, apresentados na Tabela IV.

Tabela IV
VALORES MÁXIMOS DE VELOCIDADE PARA A ABORDAGEM DE BENBOUABDALLAH *et al.*

Método	$\max(v)$ [m/s]	$\max(\omega)$ [°/s]
SpeedyGA	1.20	300.00
fmincon	1.20	199.00

A Figura 3 apresenta um comparativo visual das trajetórias obtidas com ambos os métodos.

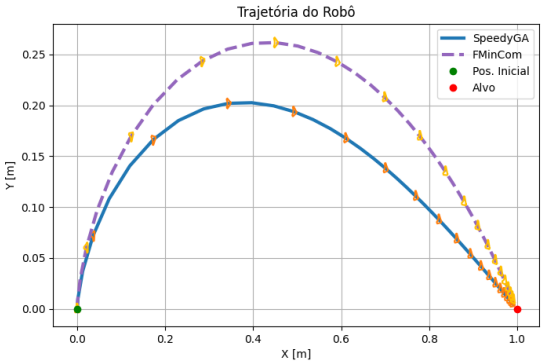

Figura 3. Trajetórias para a abordagem de Benbouabdallah *et al.*

C. Resultados - Abordagem de Breno et al.

Nesta última abordagem, utilizou-se como base para a função de ajuste aquela apresentada em [3], expressa na Equação 12.

$$F = \frac{1}{|\max(y) - y_{ref}|} + \frac{1}{|\max(v) - v_{ref}|} \tag{12}$$

Essa função tem como objetivo manter os valores máximos do deslocamento no eixo y $(\max(y))$ e da velocidade linear $(\max(v))$ o mais próximos possível de seus valores de referência: $v_{ref} = 1.2\,\text{m/s}$ e $y_{ref} = 0.15\,\text{m}$, durante toda a simulação da cena.

Entretanto, essa formulação não leva em consideração a restrição da velocidade angular. Por esse motivo, ela foi modificada com a inclusão de um termo adicional, de forma a fazer com que os métodos de otimização também considerassem essa variável. Assim, a função de ajuste utilizada passou a ser a expressa na Equação 13.

$$F = \frac{1}{|\max(y) - y_{ref}|} + \frac{1}{|\max(v) - v_{ref}|} + \frac{1}{|\max(\omega) - \omega_{ref}|} \tag{13}$$

Consequentemente, a função de custo (J) utilizada foi definida como o oposto da função de ajuste, conforme a Equação 14.

$$J = -F \tag{14}$$

Com as funções estabelecidas, os algoritmos foram executados. Para o algoritmo genético, adotou-se uma população de 200 indivíduos, distribuídos em 40 gerações, com probabilidade de mutação de 3% e probabilidade de *crossover* de 100%.

A execução dos métodos de otimização resultou nos ganhos da lei de controle proposta, apresentados na Tabela V.

Tabela V
GANHOS OBTIDOS PARA A ABORDAGEM DE BRENO *et al.*

Método	τ	κ
SpeedyGA	0.97	3.24
fmincon	1.21	3.96

Simulando o modelo com esses parâmetros, obtiveram-se os valores máximos das variáveis de interesse, apresentados na Tabela VI.

Tabela VI
VALORES MÁXIMOS PARA A ABORDAGEM DE BRENO *et al.*

Método	$\max(y)$ [m]	$\max(v)$ [m/s]	$\max(\omega)$ [°/s]
SpeedyGA	0.15	0.76	291.83
fmincon	0.15	0.93	356.32

A Figura 4 apresenta a comparação entre as trajetórias obtidas para ambos os métodos durante a simulação.

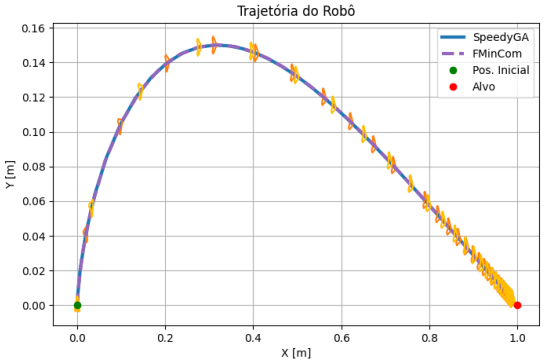

Figura 4. Trajetórias para a abordagem de Breno *et al.*

V. CONCLUSÃO

A partir dos resultados obtidos com os algoritmos de otimização, observa-se que os ganhos determinados por cada método apresentam certa discrepância em seus valores. Tal fato, entretanto, não constitui necessariamente um problema, pois existem diversas combinações de ganhos capazes de produzir desempenhos equivalentes.

Ao analisar as tabelas de valores máximos, verifica-se que os ganhos encontrados mantiveram o comportamento do modelo dentro dos limites físicos impostos, com exceção da otimização via *fmincon* aplicada à abordagem de Breno *et al.*, que excedeu o limite de velocidade angular.

O bom desempenho geral dos algoritmos está diretamente relacionado à adequada escolha das funções objetivo, as quais induziram a consideração das restrições físicas durante o processo de otimização.

Com exceção do caso mencionado para a abordagem de Breno *et al.*, os algoritmos mostraram-se equivalentes em eficácia. Embora, ao se utilizar o *SpeedyGA*, os ganhos obtidos proporcionassem um tempo menor para o robô alcançar o alvo, tal diferença não se traduz em melhorias significativas em relação ao método tradicional.

Por outro lado, o algoritmo genético demanda maior tempo de execução em comparação ao método baseado em gradiente. Contudo, quando a otimização é realizada de forma *offline*, essa diferença temporal não impacta de maneira relevante o uso prático.

Dessa forma, a escolha entre um ou outro algoritmo deve ser feita pelo projetista, levando em consideração fatores como tempo de execução, robustez frente a mínimos locais e familiaridade com as ferramentas de otimização.

REFERÊNCIAS

[1] M Aicardi, G Casalino, A Biechi, and A Balestrino. Closed loop steering of unicycle-like vehicles via. 1995.

[2] Karim Benbouabdallah and Zhu Qi-Dan. Research article improved genetic algorithm lyapunov-based controller for mobile robot tracking a moving target. *Research Journal of Applied Sciences, Engineering and Technology*, 5(15):4023–4028, 2013.

[3] Breno P de Meneses, Gabriel HV da Silva, Lara R Sobral, Mateus S Marques, Rodrigo T de Araujo, and Antonio MN Lima. Navigation of a two-wheel differential drive robot in a partially unknown environment. In *Simpósio Brasileiro de Automação Inteligente-SBAI*, volume 1, 2023.

[4] SciPy Community. scipy.optimize.minimize – minimization of scalar function of one or more variables. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html, 2025. Acessado em 06 de Agosto de 2025.
[5] Keki Burjorjee. Speedyga: A fast simple genetic algorithm. https://www.mathworks.com/matlabcentral/fileexchange/15164-speedyga-a-fast-simple-genetic-algorithm, 2025. MATLAB Central File Exchange.