OTIMIZAÇÃO DO CONTROLE DE MOTORES CC POR REDES NEURAIS

NEVES, Vinicius Aquino[1]
SILVA JR., Dalmo Cardoso da[2]
ABRITTA, Camila do Carmo Almeida[3]
PICCININI, Marco Aurélio[4]

Linha de pesquisa: Automação

RESUMO

Este trabalho de conclusão de curso investiga a otimização do controle de motores de corrente contínua (CC) por meio de Redes Neurais Artificiais (RNAs), com foco na determinação dos ganhos de controladores PID. O estudo visa comparar a eficácia das RNAs em ajustar automaticamente os parâmetros do controlador PID com métodos tradicionais de sintonia manual. Para isso, um modelo de RNA será desenvolvido e treinado para otimizar os ganhos do PID, enquanto um modelo de referência manual será criado utilizando técnicas convencionais. A comparação dos resultados se concentrará na análise do desempenho do motor CC, na precisão e na resposta dinâmica dos controladores. Este estudo busca demonstrar as vantagens das RNAs em termos de eficiência e adaptabilidade no controle de sistemas complexos, proporcionando avanços significativos na automação industrial e no desenvolvimento de tecnologias de controle mais inteligentes e autônomas.

Palavras-chave: Motores de corrente contínua. Controladores PID. Redes neurais artificiais. Otimização de controle. Automação.

[1] Graduando em Engenharia Elétrica pelo Centro Universitário Academia - UniAcademia.
[2] Professor do curso de Engenharia Elétrica do Centro Universitário Academia - UniAcademia.
[3] Professor do curso de Engenharia Elétrica do Centro Universitário Academia - UniAcademia.
[4] Professor do curso de Engenharia Elétrica do Centro Universitário Academia - UniAcademia.

UniAcademia
Centro Universitário

1 INTRODUÇÃO

Os motores de corrente contínua (CC) são amplamente utilizados em aplicações industriais e acadêmicas, especialmente em automação, acionamentos elétricos e robótica, devido à sua simplicidade, baixo custo e facilidade no controle de rotação e torque (Ogata, 2010). Dentre os métodos de controle, o controlador proporcional-integral-derivativo (PID) se destaca por sua simplicidade e bom desempenho em diversas aplicações (Astrom, 1995).

A eficácia de um controlador PID depende da escolha correta de seus parâmetros — os ganhos K_p, K_i e K_d. No entanto, definir esses valores manualmente nem sempre garante um desempenho ideal, especialmente em sistemas com variações de carga, ruídos e incertezas nos parâmetros do modelo. Métodos tradicionais, como *Ziegler-Nichols* ou tentativa e erro, muitas vezes não asseguram estabilidade e desempenho ótimo simultaneamente (Nise, 2012).

Neste contexto, técnicas de Inteligência Artificial (IA), como Redes Neurais Artificiais (RNAs), destacam-se como ferramentas eficazes para otimização e controle adaptativo. As RNAs, inspiradas no cérebro humano, aprendem relações complexas a partir de dados, sem exigir modelos matemáticos precisos (Haykin, 2001). Quando aplicadas à sintonia de controladores PID, as RNAs podem mapear a relação entre os parâmetros do controlador e o desempenho do sistema, promovendo um ajuste mais eficiente e robusto (Söderstrom, 1989).

Este trabalho propõe a otimização *off-line* do controle de rotação de um motor CC usando uma RNA, treinada para prever os valores ideais dos ganhos PID. Um modelo do sistema foi implementado em MATLAB/*Simulink*, testando várias combinações de ganhos K_p, K_i e K_d, avaliadas por uma função de custo que considera métricas como erro quadrático médio e *overshoot* da resposta do motor. Os dados obtidos treinam a RNA para estimar os parâmetros ótimos que minimizam o erro e melhoram a estabilidade do sistema.

UniAcademia
Centro Universitário

2 REVISÃO DA LITERATURA

2.1 HISTÓRICO E EVOLUÇÃO DOS CONTROLADORES DE MOTORES CC

A trajetória dos controladores de motores CC ilustra o avanço contínuo das tecnologias de controle e a busca por sistemas mais eficientes e adaptáveis. No início, o controle era feito manualmente ou por reguladores mecânicos simples, o que oferecia pouca precisão e exigia frequente intervenção do operador. Com o advento dos controladores analógicos no início do século XX, houve uma melhoria significativa em termos de precisão e estabilidade, possibilitando ajustes automáticos baseados no *feedback* dos sistemas (Ogata, 2010).

O surgimento dos controladores PI (Proporcional-Integral) e PID (Proporcional-Integral-Derivativo) foi um divisor de águas na indústria, oferecendo uma abordagem mais robusta para lidar com variações de carga e distúrbios externos, além de simplificar o projeto dos sistemas de controle (Ogata, 2010). Apesar disso, os controladores tradicionais enfrentam desafios em sistemas não lineares e dinâmicos, que são frequentemente encontrados nas aplicações industriais modernas.

Com o avanço da computação digital e o aumento da capacidade de processamento, os controladores digitais começaram a substituir os analógicos, proporcionando maior flexibilidade e a capacidade de implementar algoritmos de controle mais complexos. Essa evolução levou ao desenvolvimento e aplicação de redes neurais artificiais (RNAs) nos sistemas de controle de motores CC. As RNAs conseguem aprender e se adaptar a condições variáveis e não lineares, capturando relações dinâmicas complexas sem depender de modelos matemáticos precisos (Haykin, 2001).

Na era da Indústria 4.0, a integração de sistemas inteligentes e conectados é essencial. Os controladores de motores CC baseados em RNAs estão se tornando fundamentais para atender às crescentes demandas por eficiência, adaptabilidade e conectividade nos processos industriais. As RNAs possibilitam que os sistemas de controle respondam de forma eficaz em ambientes onde as características dos sistemas são incertas ou mudam frequentemente, consolidando seu papel na automação moderna.

2.2 AVANÇOS EM REDES NEURAIS ARTIFICIAIS PARA CONTROLE

Os avanços RNAs representam uma verdadeira revolução no campo da automação e do aprendizado de máquinas, abrindo novas perspectivas para o gerenciamento de sistemas complexos e dinâmicos. Inicialmente inspiradas no funcionamento do cérebro humano, as RNAs evoluíram de simples perceptrons para arquiteturas multicamadas sofisticadas, capazes de captar padrões não lineares em vastos conjuntos de dados. Com o tempo, técnicas de aprendizado profundo, como redes neurais convolucionais (CNNs) e redes neurais recorrentes (RNNs), ampliaram suas capacidades, permitindo um processamento mais eficaz de informações temporais e espaciais. Segundo Haykin (2001), "as redes neurais artificiais têm a capacidade de modelar relações complexas entre entradas e saídas, o que as torna adequadas para uma ampla gama de aplicações".

No contexto do controle de sistemas, as RNAs destacam-se por sua habilidade em modelar e prever comportamentos de sistemas não lineares, possibilitando a implementação de controladores mais adaptativos e eficientes (Suykens, 1995). A integração de algoritmos de aprendizado por reforço com redes neurais tem viabilizado a criação de sistemas de controle que não apenas reagem a mudanças ambientais, mas também aprendem a otimizar suas ações para atingir objetivos específicos (Perrusquía, 2021). Técnicas de treinamento avançadas, como *backpropagation* aprimorado e otimizadores adaptativos, aprimoram a precisão e a velocidade de aprendizado das RNAs, tornando-as indispensáveis em aplicações de tempo real (Rumelhart, 1986).

A aplicação de RNAs em sistemas de controle abrange desde o ajuste fino de processos industriais até a automação de veículos autônomos, que demandam alta precisão e segurança. As RNAs não apenas otimizam o desempenho dos sistemas de controle, mas também oferecem robustez superior frente a distúrbios e incertezas, consolidando seu papel como componentes essenciais na nova era da automação inteligente e conectada.

2.3 COMPARAÇÃO ENTRE CONTROLADORES TRADICIONAIS E RNAs

A comparação entre os controladores clássicos, como PI e PID, e as RNAs no contexto de sistemas de controle revela diferenças importantes em abordagem,

capacidade adaptativa e aplicabilidade. Os controladores tradicionais são amplamente utilizados na indústria pela sua simplicidade, baixo custo computacional e eficiência em sistemas lineares. Entretanto, sua eficácia depende diretamente da correta escolha dos ganhos, que muitas vezes precisam ser ajustados manualmente ou com base em métodos empíricos (Ogata, 2010). Essa limitação se torna mais evidente em sistemas com dinâmicas não lineares, variações de carga ou incertezas estruturais, onde a resposta do controlador pode se tornar insatisfatória.

As RNAs, por sua vez, representam uma abordagem mais flexível e inteligente para controle de sistemas dinâmicos. Elas não necessitam de um modelo matemático preciso e são capazes de aprender diretamente a partir dos dados, capturando características não lineares e comportamentos complexos. Durante o processo de treinamento, seus pesos e bias são ajustados para minimizar um critério de erro, permitindo que a rede generalize e se adapte a diferentes condições de operação. Essa capacidade de aprendizagem e adaptação torna as redes neurais especialmente úteis em aplicações onde os métodos clássicos se mostram insuficientes (Haykin, 2001).

Apesar de exigirem maior poder computacional e um conjunto de dados adequados para o treinamento, as RNAs vêm se tornando cada vez mais acessíveis devido ao avanço das tecnologias de hardware e algoritmos de otimização. Assim, embora os controladores clássicos ainda sejam eficazes para sistemas simples e estáticos, as RNAs oferecem uma alternativa poderosa e promissora para o controle de sistemas modernos, dinâmicos e interconectados.

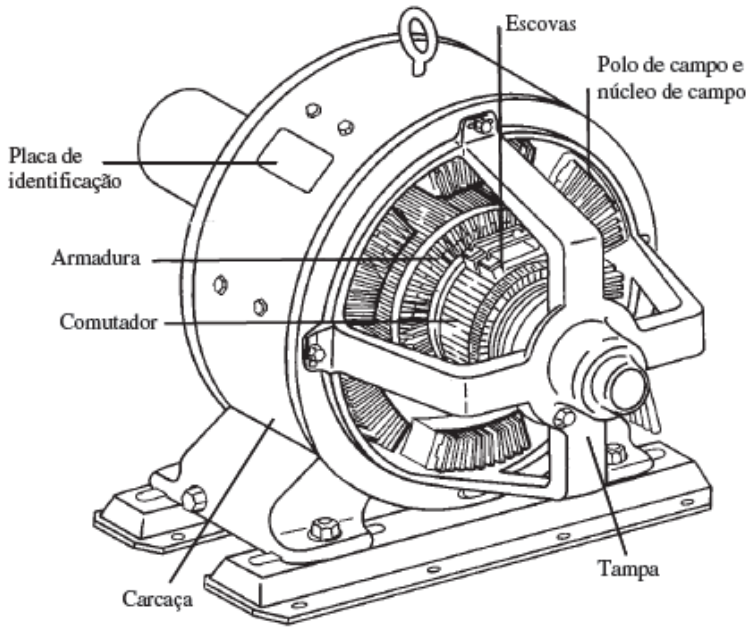
3 FUNDAMENTAÇÃO TEÓRICA

3.1 PRINCÍPIOS DE OPERAÇÃO DOS MOTORES CC

Os motores de corrente contínua (CC) são amplamente utilizados em sistemas que requerem controle preciso de velocidade e torque. Seu funcionamento baseia-se na interação entre campos magnéticos e corrente elétrica, promovendo a conversão de energia elétrica em energia mecânica, conforme os princípios da indução eletromagnética descritos por Faraday (Nunes, 2017).

Nos motores CC, a corrente elétrica é conduzida ao enrolamento do rotor por meio de um conjunto de escovas e um comutador. Este sistema assegura que o fluxo de corrente seja constantemente redirecionado, mantendo o torque na mesma direção durante o movimento rotacional. O campo magnético necessário é gerado por ímãs permanentes ou eletroímãs no estator, e a interação entre este campo e o campo gerado pela corrente no rotor resulta na força que faz o rotor girar (Chapman, 2013), como mostra a Figura 1.

FIGURA 1 - Diagrama simples de um motor CC



Fonte: Chapman, 2013.

Além de oferecer controle de rotação preciso, os motores CC são reconhecidos pela rápida capacidade de resposta, características que são altamente valorizadas em robótica, veículos elétricos e sistemas de acionamento industrial. Fatores como resistência interna, perda por atrito nas escovas e a qualidade do comutador podem influenciar a eficiência e o desempenho do motor, tornando essencial o projeto cuidadoso e a manutenção regular para garantir uma operação eficaz e duradoura.

Compreender os princípios de operação dos motores CC é fundamental não apenas para sua aplicação em sistemas de controle modernos, mas também para o

desenvolvimento de soluções inovadoras que integrem tecnologias emergentes, como redes neurais, para otimizar seu desempenho em diversas aplicações.

3.2 CONTROLADORES PID

Os controladores PID (Proporcional-Integral-Derivativo) são pilares nos sistemas de controle industrial, amplamente adotados devido à sua simplicidade, robustez e eficácia em diversas aplicações, como controle de motores, sistemas térmicos e processos químicos. Fundamentados em uma abordagem linear, os controladores PID operam sob a suposição de que o sistema a ser controlado pode ser descrito por uma função de transferência linear invariante no tempo. Essa premissa facilita o projeto e a análise utilizando ferramentas clássicas da engenharia de controle (Ogata, 2010).

Entretanto, a hipótese de linearidade constitui uma simplificação que raramente representa com exatidão o comportamento de sistemas físicos reais, os quais comumente exibem características não lineares. Em motores CC, por exemplo, tais não linearidades podem estar associadas ao atrito estático, saturação magnética, zona morta em atuadores, folgas mecânicas (*backlash*) e variações na resistência dos enrolamentos devido ao aquecimento durante a operação (Nise, 2015; Dorf; Bishop, 2011). Essas particularidades dificultam a modelagem precisa do sistema e comprometem o desempenho de controladores clássicos, como o PID, podendo ocasionar oscilações, atrasos na resposta e erros em regime permanente (Ogata, 2010).

Apesar dessas limitações, os controladores PID continuam populares devido à facilidade de implementação e sintonia. No entanto, sua aplicação em sistemas reais frequentemente exige técnicas adicionais, como compensações não lineares, controle adaptativo ou a integração com métodos de inteligência artificial, como RNAs, para melhorar a adaptabilidade e o desempenho diante de variações dinâmicas do sistema. Assim, enquanto os controladores PID são projetados com base em modelos lineares, sua aplicação eficaz em ambientes reais requer uma análise criteriosa das limitações impostas pelas não linearidades inerentes aos sistemas físicos (Astrom, 2010).

Na equação 1 podemos observar a fórmula geral do controle PID:

$$u(t) = K_p e(t) + K_i \int e(t)\, dt + K_d \frac{de(t)}{dt} \quad (1)$$

Parâmetros do controlador:

O K_p - Ganho proporcional, controla a intensidade da resposta proporcional ao erro. Quanto maior o erro, maior a ação de controle e aumenta a rotação de resposta, mas pode causar *overshoot* (ultrapassagem).

O K_i - Ganho integral, soma o erro ao longo do tempo e serve para eliminar o erro estacionário (*offset*). Pode tornar o sistema mais lento e instável se for alto demais.

O K_d - Ganho derivativo, atua com base na variação do erro (derivada) e ajuda a reduzir o *overshoot* e a melhorar a estabilidade. Muito sensível a ruídos se mal ajustado.

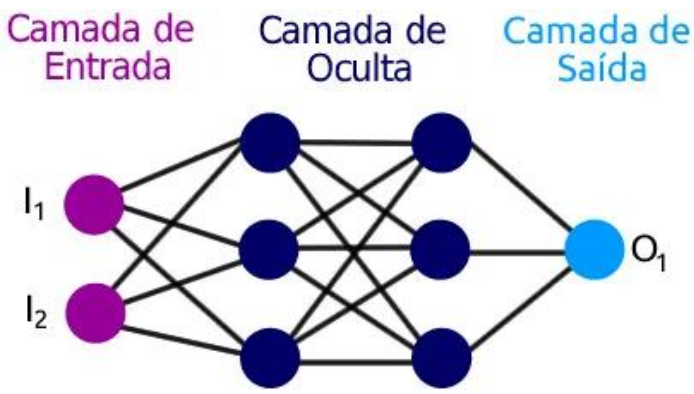
O $u(t)$ é a saída do controlador (sinal de controle).

E o $e(t)$, é o erro que é a diferença entre referência e saída real $(r(t) - y(t))$.

3.3 ARQUITETURA E FUNCIONAMENTO DE REDES NEURAIS ARTIFICIAIS

A arquitetura e funcionamento das RNAs são compreendidos através de uma estrutura em camadas (Figura 2): a camada de entrada, que recebe os dados iniciais; as camadas ocultas, responsáveis por extrair características progressivamente mais complexas; e a camada de saída, que fornece a predição final.

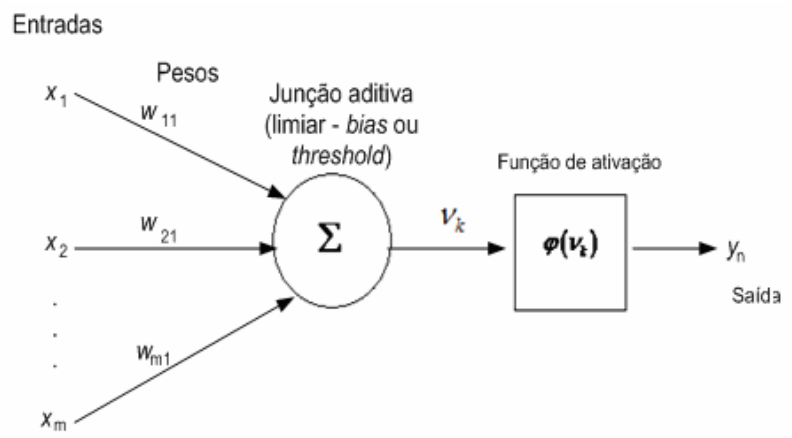
FIGURA 2 - Modelo de uma Rede Neura Artificial - Estrutura multicamada



Fonte: Grübler, 2005.

Cada unidade, ou neurônio como mostra a Figura 3, processa uma combinação linear das entradas ponderadas, adiciona um bias, e aplica uma função de ativação, como sigmoide, ReLU ou tangente hiperbólica, introduzindo não linearidade ao sistema. Essa estrutura permite à rede modelar relações complexas além do plano linear.

FIGURA 3 - Modelagem de neurônio artificial



Fonte: Voigt, Marangoni, 2024.

Os sinais de entrada $(x_1, x_2, \ldots, x_n)$ são informações que alimentam o neurônio artificial, como valores de sensores ou características de imagens. Cada entrada é multiplicada por um peso sináptico $(w_1, w_2, \ldots, w_n)$, determinando sua importância relativa no processo de decisão; pesos mais altos indicam maior influência na saída final. O somatório (Σ) acumula os produtos entre entradas e pesos, formando um valor intermediário. Para maior flexibilidade, adiciona-se um bias (b), que desloca o ponto de ativação da função, permitindo ajustes mesmo quando os sinais de entrada são nulos.

O resultado do somatório com o bias é o potencial de ativação (v), a estimativa interna do neurônio antes do processo de decisão. Este potencial é transformado por uma função de ativação (φ), que introduz não linearidade à rede, essencial para modelar sistemas complexos. Funções comuns incluem a sigmoide, ReLU e tangente hiperbólica. Após a ativação, gera-se o sinal de saída (y), interpretado como predição, classificação ou valor contínuo, dependendo da aplicação. Este sinal é então

encaminhado para a próxima camada ou utilizado diretamente na tarefa de interesse. Esses elementos, embora simples individualmente, formam a base da computação neural, essenciais para a capacidade da rede em aprender e generalizar a partir de exemplos. As equações matemáticas que modelam um neurônio podem ser formalmente representadas pelas equações 2 e 3.

$$u_k = \sum_{j=1}^{n} w_{kj} x_j \tag{2}$$

$$y_k = \varphi(u_k + b_k) \tag{3}$$

Onde:

- Os sinais de entrada estão representados pelo vetor x;
- O vetor w representam os pesos sinápticos do neurônio k e o vetor y;
- u_k é a saída do somatório;
- O bias é representado por b_k;
- A função de ativação é representada por φ;

As redes neurais envolvem diversos conceitos fundamentais que são essenciais para a compreensão de seu funcionamento e aplicação em diferentes áreas.

Feedforward é um tipo de rede neural onde as informações se movimentam em apenas uma direção: das camadas de entrada, passando por eventuais camadas ocultas, até a camada de saída. Essa arquitetura não possui ciclos, o que facilita o processamento e a análise dos dados (Bailer, 2001).

Overfitting ocorre quando um modelo aprende muito bem os detalhes e ruídos do conjunto de dados de treinamento, a ponto de comprometer sua capacidade de generalizar para dados novos. Isso significa que o modelo fica "ajustado demais" aos dados de treinamento, resultando em previsões imprecisas para dados que não foram vistos (Chakraborty, Dong, Stone, 2021).

Épocas referem-se ao número de passagens completas sobre o conjunto de dados de treinamento durante o aprendizado de uma rede neural. Em cada época, a

rede ajusta seus pesos com base nos erros identificados, visando minimizar a diferença entre as previsões e os valores reais (Chollet, 2018).

O Erro Quadrático Médio (MSE) é uma medida estatística usada como função de perda, que calcula a média dos quadrados das diferenças entre os valores previstos pelo modelo e os valores reais. Essa métrica é amplamente utilizada para avaliar a precisão do modelo (Chollet, 2018).

O Erro Quadrático Médio da Previsão (RMSE) é uma extensão do MSE, que fornece a raiz quadrada do MSE. O RMSE é útil para apresentar o erro nas mesmas unidades que os dados originais, facilitando a interpretação da precisão das previsões (Chollet, 2018).

3.4 MODELAGEM MATEMÁTICA DO MOTOR CC

Os motores CC desempenham um papel crucial na indústria, sendo amplamente utilizados para o controle eficiente de processos devido à sua estabilidade e à capacidade de ajustar com precisão a rotação, o torque e a posição. A modelagem desses motores é geralmente realizada sob a forma de sistemas SISO (*Single-Input Single-Output*), nos quais a rotação é controlada por meio da tensão aplicada à armadura do motor (Nagarathnam, Deepa, 2019).

Comumente, modelos lineares de segunda ordem são empregados para representar o comportamento dos motores CC. No entanto, esses modelos frequentemente não levam em consideração o atrito de Coulomb, o que pode resultar em zonas mortas e complicar o controle, especialmente em operações com baixas rotações. Para enfrentar esses desafios, são utilizadas técnicas de controle avançadas, que requerem uma modelagem matemática sólida e abrangente (Astrom, 2010).

A representação eletromecânica do sistema é de suma importância para o desenvolvimento de controladores PID. Esses controladores são essenciais para garantir a precisão e a eficácia do sistema de controle, permitindo ajustes finos que são fundamentais para o funcionamento otimizado dos motores (Nise, 2015). A modelagem matemática adequada é, portanto, um passo fundamental no projeto e implementação de sistemas de controle eficazes. A equação 4 descreve o comportamento elétrico do motor CC:

$$V_a(t) = R_a i_a + L_a \frac{di_a(t)}{dt} + K_e \omega(t) \quad (4)$$

Onde:

- $V_a(t)$: Tensão aplicada ao motor (V)
- R_a: Resistência da armadura (Ω)
- $i_a(t)$: Corrente da armadura (A)
- L_a: Indutância da armadura (H)
- K_e: Constante de força contra eletromotriz ($V \cdot s/rad$)
- $\omega(t)$: Velocidade angular do eixo do motor (rad/s)

Resumo: A tensão aplicada ao motor é usada para vencer a queda de tensão ôhmica, a reatância indutiva e a tensão induzida pela rotação do motor (força contra eletromotriz).

A equação 5 descreve o comportamento mecânico do motor CC:

$$J \frac{d\omega(t)}{dt} = K_t i_a(t) - B\omega(t) \quad (5)$$

Onde:

- J: Momento de inércia do rotor ($kg \cdot m^2$)
- B: Coeficiente de atrito viscoso ($N \cdot m \cdot s$)
- $\omega(t)$: Velocidade angular do eixo do motor (rad/s)
- K_t: Constante de torque ($N \cdot m/A$)
- $i_a(t)$: Corrente da armadura (A)

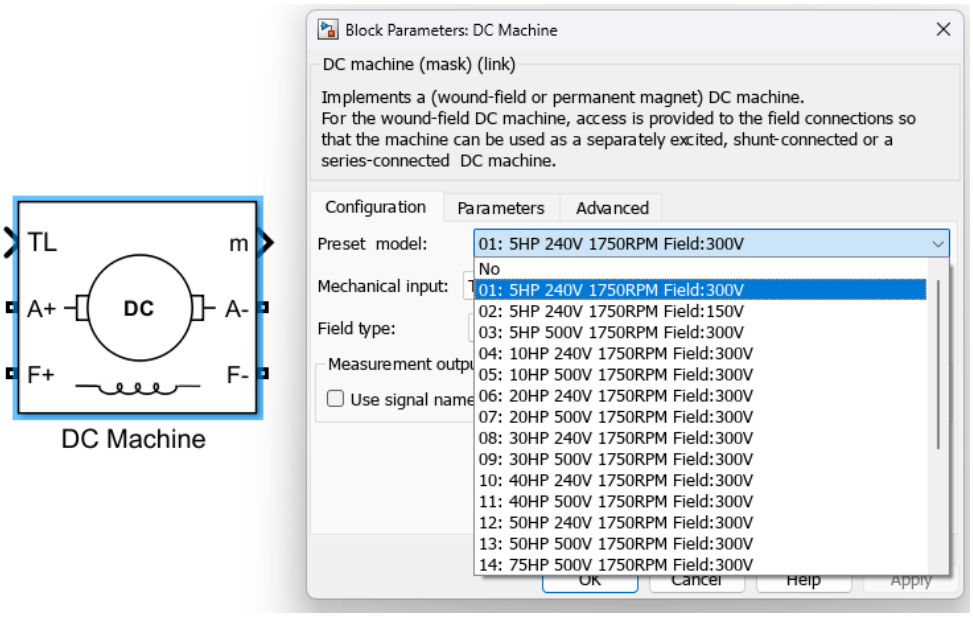
Resumo: O torque gerado pelo motor é usado para acelerar o rotor (inércia) e vencer o atrito viscoso.

4 DESENVOLVIMENTO

4.1 MOTOR CC SELECIONADO

Para o desenvolvimento do presente trabalho, foi adotado um modelo de motor disponibilizado nativamente pelo ambiente de simulação *Simulink*, da *MathWorks*. A escolha desse modelo visou garantir confiabilidade nos dados simulados, facilidade de integração com blocos de controle e compatibilidade com os recursos utilizados na etapa de otimização por redes neurais. O bloco selecionado foi o *DC Machine*, configurado com um dos motores disponíveis na lista suspensa do próprio *Simulink*, descrito como: 01: 05 HP 240V 1750RPM Field:300V como apresentado na Figura 4.

FIGURA 4 – Parâmetros do bloco *Simulink – DC machine/Configuration/Preset model*



Fonte: Autor, 2025.

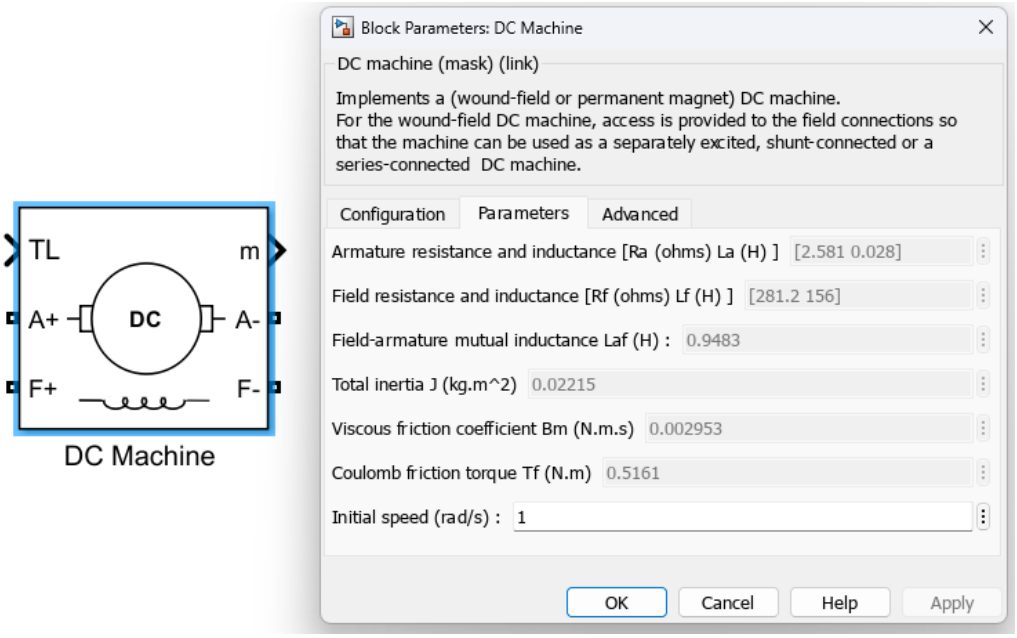
Este tipo de motor é amplamente utilizado em aplicações de controle de rotação e torque, sendo especialmente útil em projetos acadêmicos e industriais que requerem precisão e resposta dinâmica ajustável. A modelagem desse motor baseia-se nas equações diferenciais que descrevem a interação eletromecânica entre as variáveis de tensão, corrente, rotação e torque. Essa representação é adequada para simulações de controle em malha fechada, como as implementadas neste trabalho (Maranhão, 2016).

O modelo selecionado é caracterizado como um motor de excitação separada, onde a corrente do campo e da armadura são fornecidas separadamente, permitindo maior controle sobre o desempenho dinâmico do motor. Os parâmetros do motor, conforme configurados no *Simulink*, estão descritos a seguir:

- Resistência da armadura (R_a): $2{,}581\ \Omega$
- Indutância da armadura (L_a): $0{,}028\ H$
- Resistência do campo (R_f): $281{,}2\ \Omega$
- Indutância do campo (L_f): $156\ H$
- Indutância mútua armadura-campo (L_{af}): $0{,}9483\ H$
- Inércia total (J): $0{,}02215\ kg \cdot m^2$
- Coeficiente de viscosidade (b): $0{,}002953\ N \cdot m \cdot s$
- Torque de atrito de Coulomb (T_f): 0.5161

Esses parâmetros demonstrados na Figura 5 foram utilizados diretamente no bloco da *DC Machine* no *Simulink*, que realiza internamente o cálculo da equação elétrica (Equação 4) e de movimento rotacional (Equação 5).

FIGURA 5 – Parâmetros do bloco *Simulink – DC machine/Parameters*



Fonte: Autor, 2025.

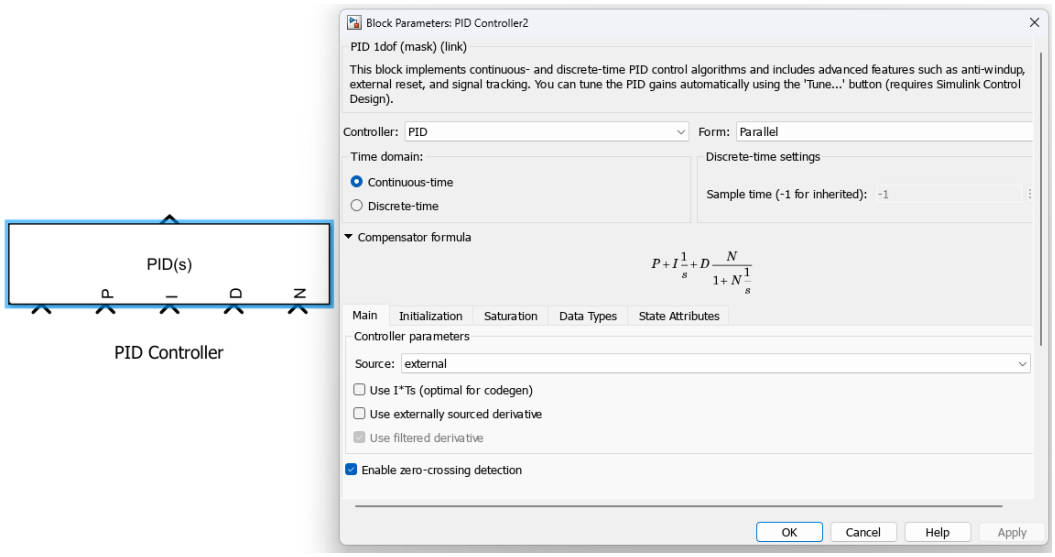
A escolha por um modelo predefinido também assegura padronização e confiabilidade, uma vez que os parâmetros já foram ajustados e testados pela própria desenvolvedora da plataforma. Isso proporciona uma base sólida para a aplicação de técnicas avançadas de controle, como a otimização por RNAs.

4.2 IDENTIFICAÇÃO DO CONTROLADOR PID

Com o objetivo de controlar a rotação do motor CC simulado no ambiente MATLAB/*Simulink*, foi inserido no sistema um controlador clássico do tipo PID, amplamente reconhecido pela sua simplicidade, robustez e eficácia em sistemas de controle industriais. O bloco utilizado para esta finalidade foi o PID *Controller*, disponível na biblioteca padrão do *Simulink*.

A configuração do bloco PID foi ajustada para operar com parâmetros externos, ou seja, os valores dos ganhos proporcional (K_p), integral (K_i), derivativo (K_d) e um filtro derivativo (N) foram definidos por variáveis externas, facilitando o processo de sintonia automática posterior utilizando uma RNA. Essa modificação foi realizada marcando a opção "*external*" nas configurações do bloco PID (Figura 6). Essa abordagem permite que os ganhos do controlador sejam atualizados dinamicamente durante as simulações, sem a necessidade de reinicialização do modelo.

FIGURA 6 – Parâmetros do bloco *Simulink* – PID *Controller*

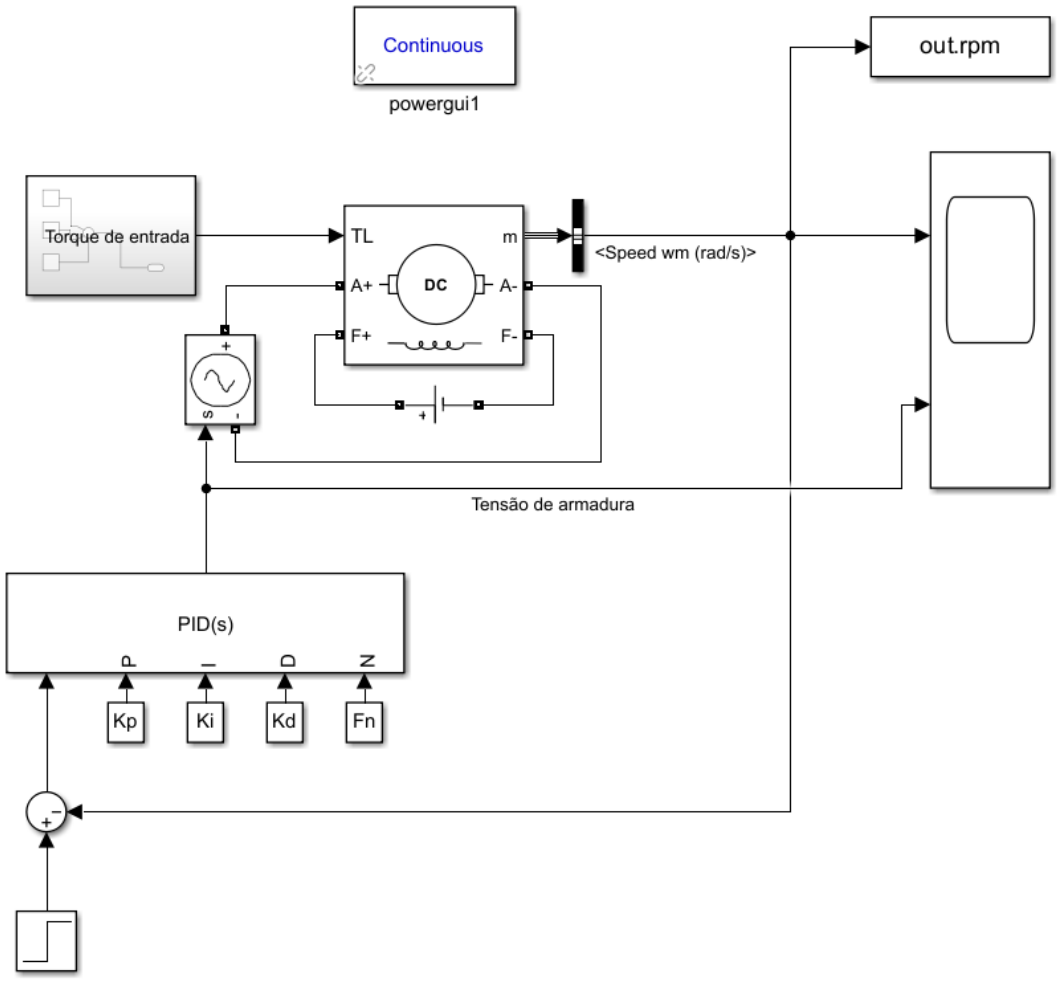


Fonte: Autor, 2025.

O controlador PID atua diretamente na tensão de entrada da armadura do motor CC, com o objetivo de regular a velocidade de rotação do eixo. O sinal de erro, calculado pela diferença entre a referência de rotação (neste projeto, 1000 rpm) e a rotação real do motor (medida por um sensor e encaminhada ao bloco *To Workspace* como *out,* rpm), é utilizado pelo controlador PID para gerar uma ação de controle proporcional, integral e derivativa.

A implementação em malha fechada possibilita que o sistema responda automaticamente a perturbações externas e variações de carga, mantendo a estabilidade e reduzindo o tempo de acomodação (Silva, 2018). A Figura 7 apresenta o diagrama de blocos da malha de controle no *Simulink*, evidenciando a inserção do controlador PID entre o somador de erro e o motor CC.

FIGURA 7 – Diagrama de blocos no *Simulink* – Malha de controle



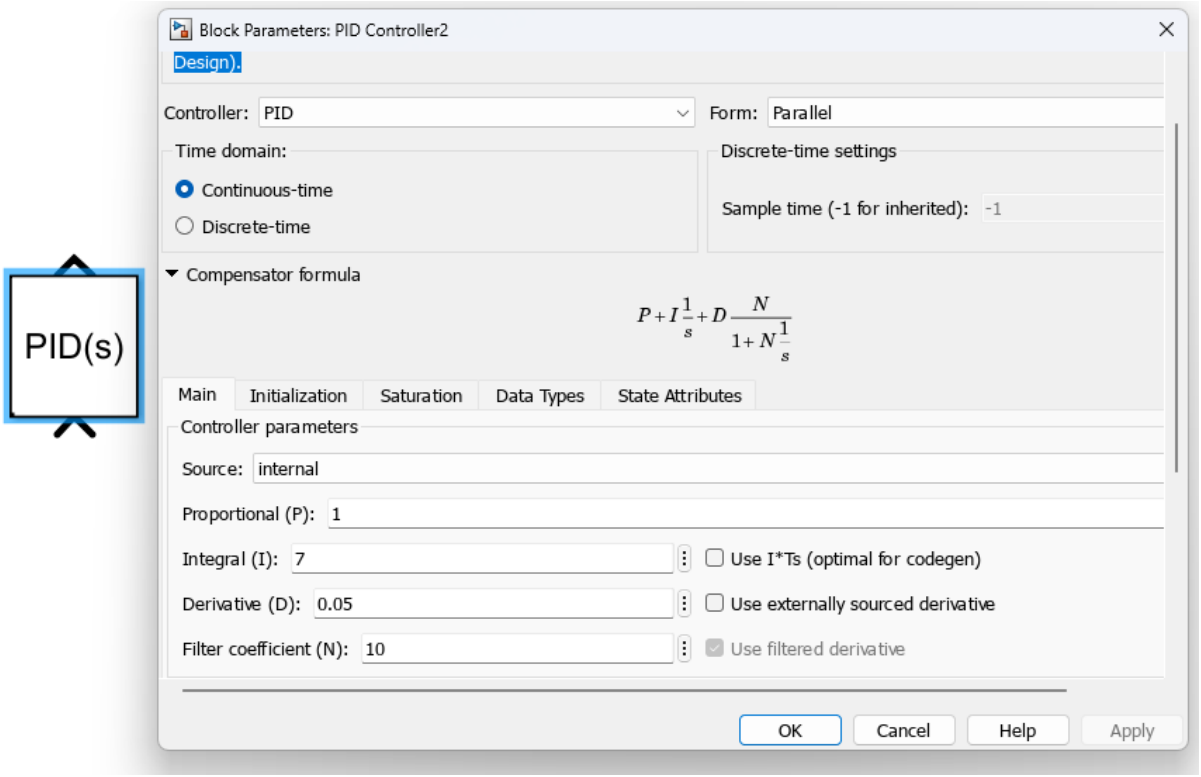
Fonte: Autor, 2025.

Inicialmente, os valores de K_p, K_i, K_d e N podem ser definidos com base em métodos de sintonia clássicos, como *Ziegler-Nichols* ou tentativa e erro. Esse trabalho inicial foi realizado manualmente utilizando o PID *Tuner* do próprio *Simulink*.

Os valores obtidos através do PID *Tuner* e posteriormente ajustados manualmente foram: $K_p = 1$, $K_i = 7$, $K_d = 0.05$ e $N = 10$.

A Figura 8 mostra a configuração selecionada para controle com PID sem utilização de rede neural para identificação dos ganhos.

FIGURA 8 – Dados selecionados para o controlador PID - Manual



Fonte: Autor, 2025.

Entretanto, como parte central deste trabalho, esses ganhos serão posteriormente trabalhados por meio de uma RNA treinada com o objetivo de encontrar de forma automática um resultado similar ou otimizado da entrada definida para a rotação.

A seguir, a resposta do sistema será avaliada com diferentes conjuntos de parâmetros para o PID, permitindo a comparação entre o desempenho obtido por

sintonia manual e pela rede neural, com foco em critérios como os picos na variação da rotação do motor e na existência de sobre-elevação (*overshoot*).

4.3 CONSIDERAÇÕES E PARÂMETROS ADOTADOS

Durante a construção do modelo de controle do motor CC no ambiente MATLAB/*Simulink*, foi fundamental considerar cuidadosamente os principais parâmetros físicos e elétricos associados à dinâmica do sistema. A adoção correta desses valores permitiu a elaboração de uma simulação realista, garantindo confiabilidade nos resultados obtidos com o controle PID e, posteriormente, com a otimização realizada por meio de RNAs.

Os parâmetros adotados impactam diretamente no desempenho da malha de controle. A inércia do sistema, por exemplo, influencia a aceleração e desaceleração do motor frente a variações de carga, afetando o tempo de resposta e a estabilidade. Já os coeficientes de atrito viscoso e de atrito de Coulomb determinam o amortecimento do sistema, contribuindo para a redução de oscilações e evitando comportamentos não lineares indesejáveis.

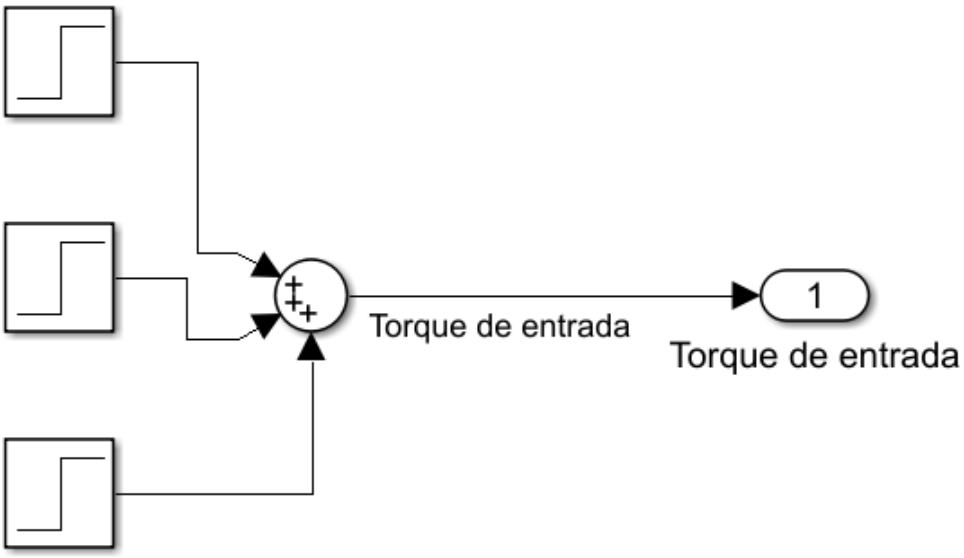
Do ponto de vista elétrico, a resistência e indutância dos enrolamentos da armadura e do campo definem a resposta eletromagnética do motor, influenciando diretamente a geração de torque (Nunes, 2017). Embora os valores exatos desses parâmetros já tenham sido apresentados no tópico anterior, é importante destacar que todos foram mantidos conforme o modelo padrão fornecido pelo bloco *DC Machine* do *Simulink,* assegurando coerência com a proposta da simulação.

Adicionalmente, para representar a excitação do campo do motor CC, foi inserido um bloco *DC Voltage Source*, configurado com uma tensão fixa de 300 V, que corresponde ao valor nominal de operação do campo. Essa configuração garante a presença de um campo magnético constante, essencial para o funcionamento adequado da máquina e para a geração de torque sob controle da armadura. A inserção deste bloco pode ser visualizada no diagrama de blocos mostrado anteriormente na Figura 7, que ilustra o subsistema de excitação do motor.

Visando simular perturbações típicas de sistemas industriais, foram adicionados três blocos do tipo *Step* responsáveis por introduzir incrementos de torque a cada 3 segundos. Cada bloco aplica um degrau de 10 N·m de carga adicional,

totalizando 30 N·m ao final do período de simulação. Essa estratégia foi adotada para avaliar o desempenho do controlador PID diante de cargas variáveis e, especialmente, testar a capacidade de adaptação da RNA em ambientes com distúrbios dinâmicos. A configuração desses blocos está representada na Figura 9, que mostra a topologia do subsistema de carga escalonada.

FIGURA 9 – Blocos para perturbação sistema no *Simulink*

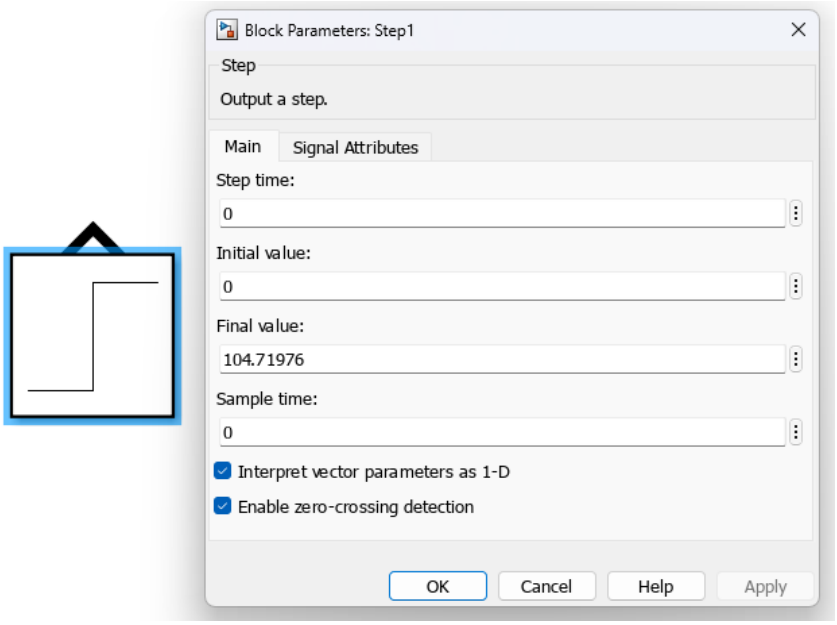


Fonte: Autor, 2025.

Essas perturbações simuladas são indispensáveis para validar a robustez do controle, tornando possível observar a estabilidade da rotação do motor, a sobre-elevação e o tempo de acomodação frente a distúrbios não previstos.

Para definir a rotação de referência, utilizou-se um bloco denominado *Step* no *Simulink*, o qual insere um valor de 1000 rpm (104.71976 rad/s) na entrada do sistema como ponto de referência como mostra a Figura 10.

FIGURA 10 – Bloco "*Step*" para entrada de valor de rotação de referência



Fonte: Autor, 2025.

4.4 DEFINIÇÃO DA FUNÇÃO DE CUSTO

A definição adequada da função de custo é um elemento essencial em qualquer processo de otimização, especialmente quando se utiliza RNAs com o objetivo de ajustar parâmetros de controladores PID. A função de custo tem a responsabilidade de quantificar, em termos numéricos, a qualidade do desempenho do sistema controlado, permitindo à rede neural aprender a minimizar os desvios indesejados em relação ao comportamento esperado (Machado, 2014).

Neste trabalho, a função de custo foi desenvolvida com base em quatro métricas principais de desempenho: Integral do Erro Absoluto (IAE), *overshoot* (OS), tempo de acomodação (Ts) e penalização por picos negativos (PN) (Machado, 2014). O objetivo é assegurar um controle que seja rápido, estável e preciso, mesmo diante de perturbações externas.

A estrutura da função de custo J é apresentada pela equação 6:

$$J = \omega_1 \cdot IAE + \omega_2 \cdot OS^2 + \omega_3 \cdot Ts + \omega_4 \cdot PN^2 \quad (6)$$

onde ω_1, ω_2, ω_3 *e* ω_4 são pesos atribuídos conforme a importância de cada termo no contexto do sistema controlado.

Na Figura 11, trecho do *script* do MATLAB com os valores considerados para os pesos:

FIGURA 11 – Parte do código da função de custo para definição dos pesos

```
% 7. Pesos (ajuste fino conforme a prioridade)
w1 = 15;   % erro acumulado
w2 = 5;  % castigo de overshoot
w3 = 3;   % tempo de acomodação
w4 = 15;  % penalização para picos negativos
```

Fonte: Autor, 2025.

Para a extração dos dados de desempenho, o modelo no *Simulink* é executado com os ganhos K_p, K_i, K_d e o filtro derivativo F_n fornecidos pela RNA. O sinal de saída da velocidade angular é comparado ao valor de referência fixado em 1000 rpm (convertido para rad/s), e as métricas são calculadas com base nesta comparação.

A IAE representa a soma total dos desvios ao longo do tempo, penalizando respostas lentas. O *overshoot* é penalizado ao quadrado para evitar ultrapassagens bruscas que comprometem a estabilidade.

As equações 7 e 8, representam os componentes IAE e OS, respectivamente:

$$IAE = \int_0^{\infty} |e(t)|\, dt \tag{7}$$

$$OS = \frac{y_{max} - r}{r} \times 100\% \tag{8}$$

onde y_{max} é o pico máximo da saída do sistema e r é o valor de referência.

O tempo de acomodação é definido como o último instante em que o erro ultrapassa 2% do valor de referência, e o termo PN penaliza quedas abruptas da rotação abaixo do valor de referência, fator crítico para sistemas com carga variável.

A formulação da função de custo neste trabalho foi fundamentada na integração de conceitos apresentados por Åström (1995) e Haykin (2001), cujas

contribuições são amplamente reconhecidas no desenvolvimento de controladores PID otimizados por métodos heurísticos e redes neurais.

4.5 BASE DE DADOS

Uma base de dados ou *dataset,* consiste em um conjunto de amostras que representam as características de um sistema físico ou simulado. Em aplicações de controle, como no ajuste de parâmetros PID via redes neurais, a base de dados deve conter entradas (ganhos K_p, K_i, K_d, F_n) e saídas (desempenho do sistema, erro residual, sobressinal, tempo de acomodação). A correta elaboração desse conjunto de informações é essencial para garantir que a rede seja capaz de prever ou otimizar os ganhos de forma eficiente (Haykin, 2009).

Para que um *dataset* seja adequado ao treinamento de redes neurais, ele deve apresentar algumas propriedades essenciais:

1. Representatividade: Os dados devem cobrir todo o espaço de busca dos parâmetros do sistema, incluindo diferentes condições operacionais e variações de carga.
2. Suficiente quantidade de amostras: O volume de dados deve ser suficiente para evitar *overfitting*, garantindo que a rede possa generalizar corretamente.
3. Distribuição equilibrada: Os exemplos devem contemplar tanto cenários típicos quanto situações extremas, permitindo que a RNA aprenda a atuar adequadamente em qualquer circunstância.
4. Consistência e precisão: Erros de medição ou registros inconsistentes podem comprometer a qualidade do treinamento, levando a previsões equivocadas.

Na implementação deste projeto, a base de dados consiste em um conjunto de pares entrada-saída apresentadas nas equações 9 e 10, sendo a entrada definida por:

$$X = [K_p, K_i, K_d, F_n] \tag{9}$$

e a saída correspondente ao custo calculado pela função de avaliação:

$$Y = função\ de\ custo(X) \tag{10}$$

Esta abordagem permite que a rede neural aprenda a relação entre os ganhos PID e a resposta dinâmica do sistema, otimizando-os para reduzir o erro e melhorar a estabilidade. A estratégia de amostragem adotada foi por geração aleatória uniforme que cobre toda a faixa de valores possíveis de maneira homogênea (Santos, 2015).

Como os parâmetros possuem ordens de grandeza distintas, é necessário aplicar técnicas de normalização para garantir que a rede neural interprete os dados corretamente. Métodos comuns incluem a equação 11 e 12:

Escalonamento Min-Max:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{11}$$

Transformação logarítmica:

$$X_{log} = log_{10}(X) \tag{12}$$

A transformação logarítmica é particularmente útil para parâmetros que variam em faixas amplas.

Foram selecionadas 1200 amostras. Além disso, foi estabelecida uma limitação que cria uma faixa na qual essas amostras podem ser geradas para cada componente. A escolha de cada faixa considerou os ganhos previamente definidos no controlador sem o uso de rede neural. Isto porque ao estabelecer uma faixa de amostras, o resultado torna-se mais confiável e exige menor custo computacional. A Figura 12 apresenta essas faixas adotadas no *script* do MATLAB.

FIGURA 12 – Parte do código de obtenção de dados – amostras/faixas

```
%% 0. CONFIGURAÇÕES BÁSICAS ----
Ntarget      = 1200;
NmaxAttempts = 1200;

% --- configurações iniciais ---
ranges = [ 0.5    20;    % Kp
           5      20;    % Ki
           0.01   0.1;   % Kd
           1      20];   % Fn
```

Fonte: Autor, 2025.

Ao final é gerado um arquivo com o nome “pid_rna_dataset.mat” que será utilizado como base de dados para treinamento da rede neural.

4.6 PROJETO E TREINAMENTO DA REDE NEURAL

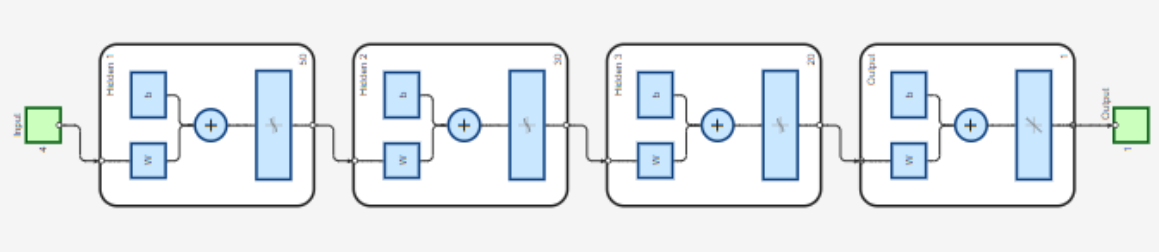
O primeiro passo no desenvolvimento de uma rede neural é preparar o conjunto de dados. Neste projeto, foram utilizados dois arquivos de dados: pid_rna_dataset.mat e pid_rna_dataset_simplificado.mat, que foram gerados pelo *script* de *dataset* O MATLAB oferece funções eficientes para a manipulação de arquivos.mat, como exist() para verificar a existência dos arquivos e load() para carregar os dados necessários. A verificação das variáveis Xlog e Y garantiu que os dados estavam prontos para uso, com um número mínimo de amostras para assegurar a validade do conjunto.

A divisão dos dados em conjuntos de treinamento e teste é uma etapa crucial. Utilizou-se a técnica de divisão *hold-out*, implementada com a função cvpartition do MATLAB, que facilita a divisão de dados em proporções específicas, como 70% para treinamento e 30% para teste. Este método é amplamente adotado devido à sua simplicidade e eficácia (Bishop, 2006).

A rede neural desenvolvida foi uma rede *feedforward*, criada com a função *feedforwardnet* do MATLAB. Esta função permite especificar o número de neurônios em cada camada oculta, sendo configurado aqui como [50, 30, 20], que representa 50 neurônios na primeira camada oculta, 30 na segunda e 20 na terceira (Figura 13). O

algoritmo de treinamento escolhido foi a Regularização Bayesiana (trainbr), que é eficaz em evitar *overfitting* (Mackay, 1992).

FIGURA 13 – Diagrama da rede neural com quatro camadas ocultas

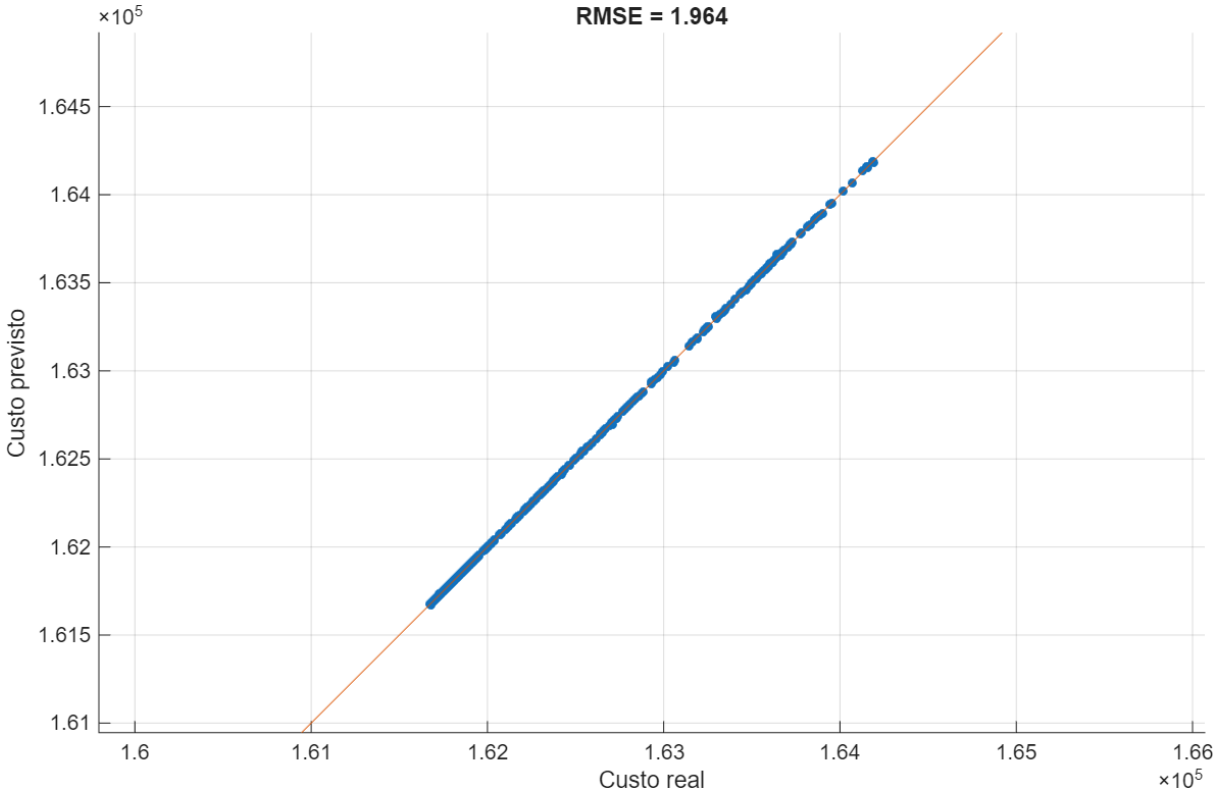


Fonte: Autor, 2025.

O treinamento da rede neural foi conduzido com um limite máximo de 1000 mil épocas, configurado por meio do parâmetro trainParam.epochs, disponível nas propriedades da rede no MATLAB. A função de desempenho adotada foi o MSE, estabelecendo-se como meta o valor de 0,4. Essa meta foi definida com base em testes sucessivos com diferentes valores, nos quais foi possível identificar a ocorrência de *overfitting* e alto custo computacional. O treinamento da rede era interrompido assim que o valor do MSE atingisse 0,4, caracterizando o ponto de convergência. Esse valor alvo do MSE foi definido por meio do parâmetro net.trainParam.goal.

Após o treinamento, o desempenho da rede foi avaliado com o conjunto de teste, calculando-se o RMSE. Esta avaliação foi facilitada pelo MATLAB, que oferece funções para calcular métricas estatísticas de forma eficiente (Figura 14).

FIGURA 14 – Representação do RMSE obtido



Fonte: Autor, 2025.

O próximo passo do *script* inicia gerando mil e duzentas combinações diferentes de parâmetros, como K_p, K_i, K_d e F_n, conhecidas como "configurações candidatas".

Para isso, são utilizadas funções básicas do MATLAB. A função rand() é utilizada para criar valores aleatórios, que geram as configurações candidatas de forma diversificada. Cada configuração é então transformada para uma escala logarítmica utilizando log10(), o que ajuda a distribuir os valores de forma mais uniforme no espaço de busca. Este passo é importante para que as combinações geradas cubram uma ampla gama de possibilidades.

Cada uma dessas configurações candidatas é avaliada pela rede neural, por meio da função net(), que estima o "custo" ou a eficácia de cada uma. A função min() é então usada para identificar a configuração que apresenta o menor custo estimado,

sendo essa selecionada como a melhor. Isso significa que a rede neural prevê que ela proporcionará o melhor desempenho no sistema de controle.

Finalmente, o *script* converte os valores logarítmicos de volta para a escala linear com 10.^ e imprime os valores dos parâmetros da configuração vencedora, utilizando fprintf(), que estão prontos para serem implementados no sistema.

5 SIMULAÇÃO E RESULTADOS

Com a etapa de treinamento da RNA concluída, procedeu-se à simulação do modelo completo no ambiente *Simulink*, considerando dois cenários distintos: um com os ganhos do controlador PID ajustados manualmente por tentativa e erro, e outro com os ganhos obtidos pela RNA treinada. O objetivo dessa comparação é avaliar a eficácia da rede neural na sintonia dos parâmetros de controle e seu impacto direto no desempenho dinâmico do motor CC.

Inicialmente, foram extraídos os melhores valores de ganhos do controlador PID produzidos pela RNA. Esses valores foram inseridos no modelo e utilizados para simular a resposta do sistema. A Tabela 1 apresenta os valores finais obtidos para os ganhos proporcional (K_p), integral (K_i) e derivativo (K_d), bem como o valor do filtro derivativo (F_n), determinado automaticamente durante o processo de otimização comparado aos valores obtidos manualmente pelo modelo com somente o PID.

TABELA 1 – Melhores valores de ganhos obtidos pela RNA x Ganhos somente com PID

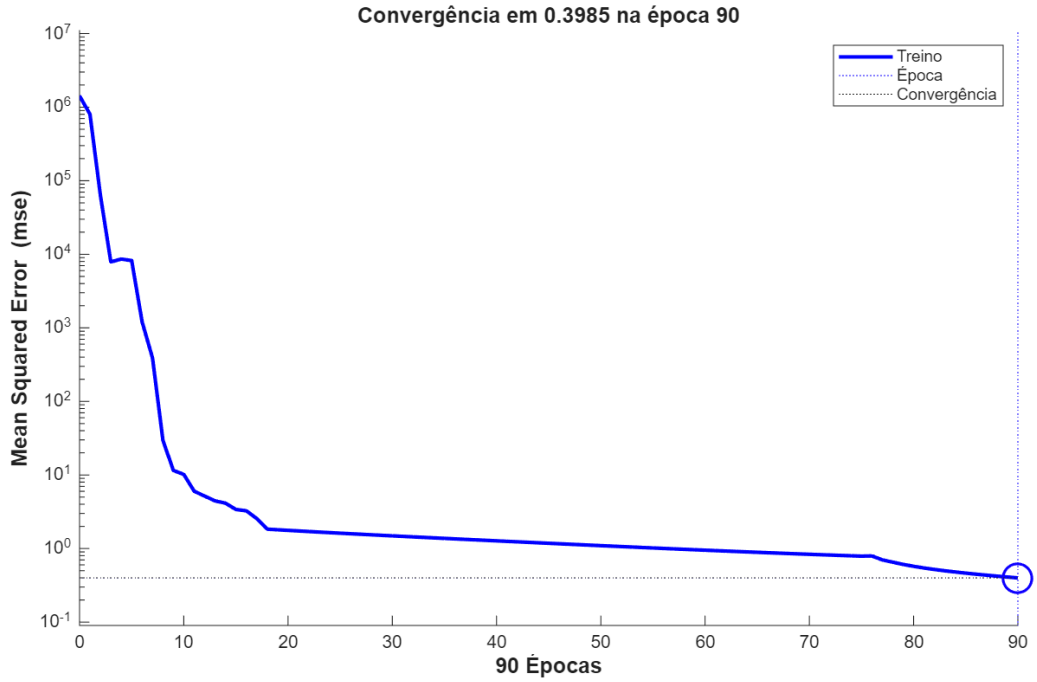
	Com RNA	Somente PID
Ganho proporcional $(\mathrm{K_p})$	2,0278	1
Ganho integral $(\mathrm{K_i})$	7,98	7
Ganho derivativo $(\mathrm{K_d})$	0,001015	0,05
Filtro derivativo $(\mathrm{F_n})$	8,867	10

Fonte: Autor, 2025.

Na máquina utilizada para o treinamento desta rede, foram necessários dois minutos e onze segundos para alcançar a convergência. O processo exigiu um total de 90 épocas até atingir o critério estabelecido, definido por um MSE igual a 0,4. Esse resultado é considerado satisfatório, tendo em vista o curto tempo de processamento

e o número reduzido de épocas, especialmente considerando que o objetivo final é controlar a rotação do motor com o mínimo possível de variações em relação ao valor de referência. (Figura 15).

FIGURA 15 – Performance do treinamento da RNA

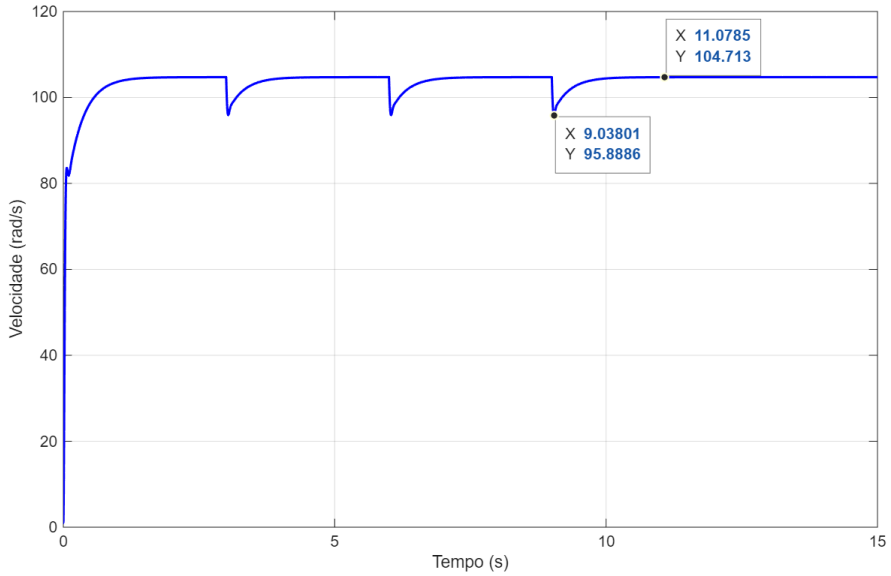


Fonte: Autor, 2025.

Em seguida, a simulação foi executada para o cenário com a rede neural ativada. O resultado obtido demonstrou uma resposta de rotação com menor variação em relação ao modelo com sintonia manual. A saída da rotação do motor foi registrada em ambos os casos para fins de comparação.

A Figura 16 mostra o gráfico da rotação do motor CC utilizando os ganhos otimizados pela RNA, indicando também os valores mínimos e máximos das variações provocadas pela entrada das cargas adicionadas em etapas.

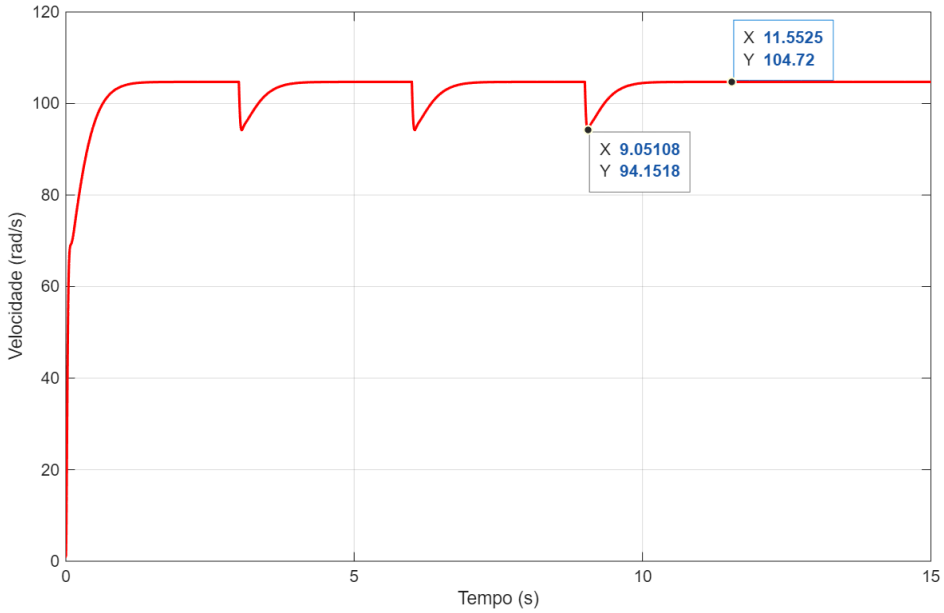
FIGURA 16 – Resposta do motor com PID otimizado por RNA



Fonte: Autor, 2025.

Logo em seguida, na Figura 17, é apresentado o gráfico correspondente ao modelo do motor CC sem o uso da RNA, utilizando apenas o controlador PID com sintonia manual.

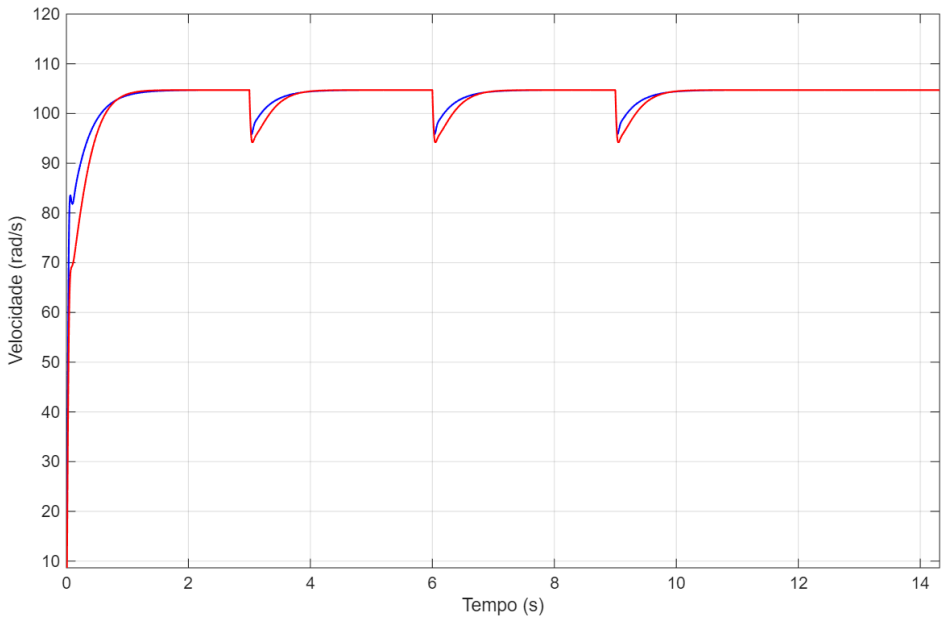
FIGURA 17 – Resposta do motor com PID manual



Fonte: Autor, 2025.

Para evidenciar com maior clareza os benefícios da abordagem com redes neurais, ambos os gráficos foram sobrepostos como mostra a Figura 18, resultando em uma figura comparativa onde se pode observar, de forma clara, a melhoria na variação da rotação, que é o principal objetivo do estudo.

FIGURA 18 – Resposta do motor com PID otimizado por RNA x PID manual



Fonte: Autor, 2025.

Ao analisarmos os gráficos de saída, é possível observar que ambos os métodos de controle apresentaram um bom desempenho em relação ao *overshoot*, com valores praticamente nulos. No que diz respeito à rotação do motor, cujo objetivo é manter-se em 1000 rpm (equivalente a aproximadamente 104,72 rad/s), ambos os sistemas conseguiram sustentar valores próximos ao desejado, mesmo após as perturbações causadas pela aplicação de cargas adicionais de torque de 10 N·m a cada 3 segundos. A Tabela 2 apresenta os valores máximos e mínimos registrados nos pontos de variação para os dois casos, evidenciando o comportamento do sistema frente às perturbações impostas.

TABELA 2 – Valores de rotação Max-Min nas oscilações

	Valor obtido		Valor de referência		Variação	
	rad/s	rpm	rad/s	rpm	rad/s	rpm
Rotação Max. (otimizado pela RNA)	104,713	999,94	104,720	1000	-0,007	-0,065
Rotação Min. (otimizado pela RNA)	95,889	915,67	104,720	1000	-8,831	-84,33
Rotação Max. (PID manual)	104,720	1000	104,720	1000	0	
Rotação Min. (PID manual)	94,150	899,06	104,720	1000	-10,57	-100,94

Fonte: Autor, 2025.

Com base nos resultados apresentados na Tabela 2, observa-se que, tanto na simulação com o controlador PID otimizado por meio de RNA, quanto na configuração com os ganhos ajustados manualmente, a velocidade do motor foi adequadamente mantida em torno do valor de referência, igual a 1000 rpm, sem ultrapassá-lo. Durante a aplicação das perturbações — representadas pelas cargas adicionais de 10 N·m inseridas em intervalos regulares — verificou-se uma breve redução na velocidade, seguida de rápida recuperação ao regime desejado.

Na simulação com o uso da RNA, a menor rotação registrada durante o intervalo de queda foi de 915,67 rpm, representando uma variação negativa de 84,33 rpm em relação ao valor de referência. Por outro lado, no modelo com controle PID ajustado manualmente, a rotação mínima observada foi de 899,06 rpm, indicando uma redução de 100,94 rpm. Ressalta-se que a diferença entre os dois métodos de controle é relevante, uma vez que a simulação com os ganhos obtidos pela RNA apresentou um desempenho superior na atenuação da variação de rotação, mantendo aproximadamente 17 rpm a mais em comparação ao modelo com ajuste manual.

6 CONCLUSÃO

O objetivo do trabalho foi investigar a aplicação de RNAs na otimização dos parâmetros de controladores PID utilizados no controle de velocidade de motores CC.

A proposta foi desenvolvida a partir de um modelo de motor CC implementado no ambiente *Simulink*, com a inclusão de um controlador PID cujos parâmetros foram ajustados por meio de dois métodos distintos: ajuste manual e otimização via RNA. Os resultados obtidos demonstraram que ambos os métodos foram eficazes em manter a rotação do motor próxima do valor de referência de 1000 rpm (104,72 rad/s), mesmo diante de perturbações introduzidas artificialmente por cargas de torque de 10 N·m aplicadas a cada 3 segundos. As simulações evidenciaram que os valores mínimos de velocidade durante as quedas foram de aproximadamente 915,67 rpm para o modelo com RNA e 899,06 rpm para o modelo com controle manual, indicando que o desempenho do controlador otimizado pela RNA foi superior.

É importante destacar que, após a configuração completa da RNA, o resultado obtido foi satisfatório, considerando que foram necessárias apenas 90 épocas, simuladas em um tempo total de dois minutos e onze segundos. Ressalta-se que a RNA foi implementada por meio do MATLAB, e que esse tempo de processamento pode variar significativamente conforme a capacidade computacional do equipamento utilizado.

Diante disso, como proposta para trabalhos futuros, sugere-se a utilização de técnicas de otimização híbridas, como algoritmos genéticos ou enxame de partículas combinados com redes neurais (Dorigo, 2004), bem como a aplicação de outros tipos de aprendizado de máquinas e redes neurais, como estratégias de *early stopping* e paralelização em *Graphics Processing Unit* (GPU) para reduzir o tempo de treinamento. Além disso, arquiteturas mais sofisticadas, como redes recorrentes (RNNs) ou Neurais Long Short Term Memory (LSTM), poderiam ser exploradas para capturar melhor as dinâmicas temporais envolvidas no processo de controle (Ogata, 2010).

Dessa forma, conclui-se que a aplicação de Redes Neurais Artificiais para otimização de controladores PID é uma alternativa viável e promissora, ainda que desafiadora. Seus benefícios devem ser analisados frente aos custos computacionais e às exigências do sistema em questão, considerando sempre a viabilidade prática de sua implementação.

ABSTRACT

This thesis investigates the optimization of direct current (DC) motor control through Artificial Neural Networks (ANNs), focusing on determining the gains of PID controllers. The study aims to compare the effectiveness of ANNs in automatically adjusting the PID controller parameters with traditional manual tuning methods. For this purpose, an ANN model will be developed and trained to optimize the PID gains, while a manual reference model will be created using conventional techniques. The comparison of results will focus on analyzing the performance of the DC motor, the precision, and the dynamic response of the controllers. The research seeks to demonstrate the advantages of ANNs in terms of efficiency and adaptability in controlling complex systems, providing significant advancements in industrial automation and the development of smarter and more autonomous control technologies.

Keywords: Direct Current Motors, PID Controllers, Artificial Neural Networks, Control Optimization, Automation.

REFERÊNCIAS

ASTROM, K. J.; HAGGLUND, T. **PID Controllers**: Theory, Design, and Tuning. 2. ed. Research Triangle Park: ISA – The Instrumentation, Systems, and Automation Society, 1995.

ASTROM, Karl J.; MURRAY, Richard M. **Feedback Systems**: An Introduction for Scientists and Engineers. 2. ed. eletrônica. Princeton: Princeton University Press, 2010.

BAILER-JONES, C. A. L.; GUPTA, R.; SINGH, H. P. **An introduction to artificial neural networks**. Cornel University, 2001.

BISHOP, Christopher M. **Pattern recognition and machine learning**. New York: Springer, 2006.

CHAKRABORTY, S.; DONG, Y.; STONE, D. J. **Overfitting, Model Tuning, and Evaluation of Prediction Performance**. In: GERO, J. S. (Ed.). Machine Learning and Artificial Intelligence in Healthcare Systems. National Center for Biotechnology Information (US), 2021. Disponível em: https://www.ncbi.nlm.nih.gov/books/NBK583970/. Acesso em: 29 jun. 2025.

CHAPMAN, Stephen J. **Fundamentos de máquinas elétricas**. 5. ed. Porto Alegre: AMGH Editora Ltda, 2013. p. 405.

CHOLLET, François**. Deep Learning with Python**. Shelter Island: Manning Publications, 2018.

CONTROL ENGINEERING. **Electronic motion control, then and now**. Publicado em 2024. Disponível em: https://www.controleng.com/electronic-motion-control-then-and-now/. Acesso em: 29 jun. 2025.

DORF, Richard C.; BISHOP, Robert H. **Sistemas de controle modernos**. 12. ed. São Paulo: Pearson, 2011.

DORIGO, Marco; STÜTZLE, Thomas. **Ant Colony Optimization**. Cambridge: MIT Press, 2004.

GRÜBLER, Murillo. **Entendendo o funcionamento de uma Rede Neural Artificial**. Medium (comunidade Brasil AI), 11 jun. 2018. Disponível em: https://medium.com/brasil-ai/entendendo-o-funcionamento-de-uma-rede-neural-artificial-4463fcf44dd0. Acesso em: 29 mai. 2025.

HAYKIN, S. **Neural Networks and Learning Machines**. 3. ed. Upper Saddle River: Pearson, 2009.

HAYKIN, S. **Neural Networks**: A Comprehensive Foundation. 2. ed. Upper Saddle River: Prentice Hall, 2001.

HAYKIN, S. **Redes neurais**: princípios e prática. 2. ed. Porto Alegre: Bookman, 2001.

MACHADO, R. C.; BERNARDES, M. A. **Controle Adaptativo e Redes Neurais Artificiais**: Fundamentos e Aplicações. São Paulo: Blucher, 2014.

MACKAY, D. J. C. **Bayesian interpolation**. Neural Computation, Cambridge, v. 4, n. 3, p. 415–447, 1992.

MARANHÃO, Marco Antônio de Souza; CAMPOS, Marcos Martins. **Modelagem e Controle de Motores Elétricos**. São Paulo: LTC, 2016.

NAGARATHNAM, K.; DEEPA, S. **Control of DC Motors**: Modeling, Simulation and Control Techniques. In: DC Motor Control: Modeling and Analysis. Springer, 2019. p. 1-40.

NISE, N. S. **Controle de Sistemas Lineares**. 6. ed. São Paulo: LTC, 2012.

NISE, Norman S. **Controle de sistemas de engenharia**. 7. ed. Porto Alegre: Bookman, 2015.

NUNES, José Carlos de Almeida. **Máquinas Elétricas**: Fundamentos, Aplicações e Simulações. 2. ed. São Paulo: Érica, 2017.

OGATA, K. **Engenharia de Controle Moderno**. 5. ed. São Paulo: Prentice Hall, 2010.

PERRUSQUÍA, Adolfo; YU, Wen. **Identification and optimal control of nonlinear systems using recurrent neural networks and reinforcement learning**: an overview. Neurocomputing, v. 438, p. 19-45, jan. 2021. Disponível em: https://doi.org/10.1016/j.neucom.2020.02.108. Acesso em: 29 jun. 2025.

RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. **Learning representations by back-propagating errors**. Nature, v. 323, n. 6088, p. 533–536, 1986. Disponível em: https://doi.org/10.1038/323533a0. Acesso em: 29 jun. 2025.

SANTOS, F. A.; SILVA, R. R. **Redes Neurais Artificiais e Aplicações em Controle**. São Paulo: Blucher, 2015.

SILVA, Cláudio G. da; FREITAS, Wilson B. de. **Controle Automático**: Teoria e Aplicações. 2. ed. São Paulo: Érica, 2018.

SÖDERSTROM, T.; STOICA, P. **System Identification**. New York: Prentice-Hall, 1989.

SUYKENS, Johan A. K.; VANDEWALLE, Joos P. L.; DE MOOR, B. L. **Artificial neural networks for modelling and control of non-linear systems**. Dordrecht: Springer, 1995.

VOIGT, F. D.; MACHADO, R. A. F.; MARANGONI, C. Desenvolvimento de uma Rede Neural Artificial para maximizar a reprodutibilidade do tingimento de tecidos de poliamida. **Revista FT**, Rio de Janeiro, v. 28, ed. 132, mar. 2024. DOI: 10.5281/zenodo.10741273. Disponível em: https://revistaft.com.br/desenvolvimento-de-uma-rede-neural-artificial-para-maximizar-a-reprodutibilidade-do-tingimento-de-tecidos-de-poliamida/. Acesso em: 29 mai. 2025.