MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO

João Pedro Aguiar Formiga Matos
Pedro Paulo Carvalho Vieira

Emprego de Inteligência Artificial na Identificação de Parâmetros Eletromagnéticos de Motores Elétricos

Goiânia - GO
2024

UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TRABALHO DE CONCLUSÃO DE CURSO DE GRADUAÇÃO NO REPOSITÓRIO INSTITUCIONAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio do Repositório Institucional (RI/UFG), regulamentado pela Resolução CEPEC no 1240/2014, sem ressarcimento dos direitos autorais, de acordo com a Lei no 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo dos Trabalhos de Conclusão dos Cursos de Graduação disponibilizado no RI/UFG é de responsabilidade exclusiva dos autores. Ao encaminhar(em) o produto final, o(s) autor(a)(es)(as) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do Trabalho de Conclusão de Curso de Graduação (TCCG)

Nome(s) completo(s) do(a)(s) autor(a)(es)(as): PEDRO PAULO CARVALHO VIEIRA e JOÃO PEDRO AGUIAR FORMIGA MATOS

Título do trabalho: **EMPREGO DE INTELIGÊNCIA ARTIFICIAL NA IDENTIFICAÇÃO DE PARÂMETROS ELETROMAGNÉTICOS DE MOTORES ELÉTRICOS**

2. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador) Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante: a) consulta ao(à)(s) autor(a)(es)(as) e ao(à) orientador(a); b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo do TCCG. O documento não será disponibilizado durante o período de embargo.

Casos de embargo:

- Solicitação de registro de patente;

- Submissão de artigo em revista científica;

- Publicação como capítulo de livro.

Obs.: Este termo deve ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Geyverson Teixeira De Paula**, **Professor do Magistério Superior**, em 22/07/2024, às 13:35, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

Documento assinado eletronicamente por **Pedro Paulo Carvalho Vieira**, **Discente**, em 29/07/2024, às 12:00, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **João Pedro Aguiar Formiga Matos**, **Discente**, em 29/07/2024, às 16:59, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4687532** e o código CRC **C633A6D2**.

Referência: Processo nº 23070.015338/2024-13 SEI nº 4687532

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO

João Pedro Formiga Matos
Pedro Paulo Carvalho Vieira

Emprego de Inteligência Artificial na Identificação de Parâmetros Eletromagnéticos de Motores Elétricos

Trabalho de conclusão de curso, apresentado à disciplina PFC, do curso de Engenharia Elétrica da Universidade Federal de Goiás como requisito parcial para obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Dr. Geyverson Teixeira de Paula

Universidade Federal de Goiás - UFG
Escola de Engenharia Elétrica, Mecânica e de Computação
Programa de Pós-Graduação em Engenharia Elétrica e de Computação

Goiânia - GO
2024

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

Aguiar Formiga Matos, João Pedro
EMPREGO DE INTELIGÊNCIA ARTIFICIAL NA IDENTIFICAÇÃO DE PARÂMETROS ELETROMAGNÉTICOS DE MOTORES ELÉTRICOS [manuscrito] / João Pedro Aguiar Formiga Matos, Pedro Paulo Carvalho Vieira. - 2024.
xxiii, 23 f.: il.

Orientador: Prof. Dr. Geyverson Teixeira de Paula.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Goiás, Escola de Engenharia Elétrica, Mecânica e de Computação (EMC), Engenharia Elétrica, Goiânia, 2024.
Bibliografia. Apêndice.
Inclui siglas, abreviaturas, símbolos, gráfico, tabelas.

1. Inteligência Artificial. 2. Parâmetros Eletromagnéticos. 3. Motores Elétricos. 4. Redes Neurais. 5. Software FEMM. I. Carvalho Vieira, Pedro Paulo. II. Paula, Geyverson Teixeira de, orient. III. Título.

CDU 621.3

UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO

ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO

Ao(s) 22 dia(s) do mês de julho do ano de 2024 iniciou-se a sessão pública de defesa do Trabalho de Conclusão de Curso (TCC) intitulado "**EMPREGO DE INTELIGÊNCIA ARTIFICIAL NA IDENTIFICAÇÃO DE PARÂMETROS ELETROMAGNÉTICOS DE MOTORES ELÉTRICOS**", de autoria de JOÃO PEDRO AGUIAR FORMIGA MATOS e PEDRO PAULO CARVALHO VIEIRA, do curso de Engenharia Elétrica, da Escola de Engenharia Elétrica, Mecânica e de Computação da UFG. Os trabalhos foram instalados pelo(a) Prof. Dr. Geyverson Teixeira de Paula (EMC/UFG) com a participação dos demais membros da Banca Examinadora: Prof. Dr. Leonardo da Cunha Brito (EMC/UFG) e Me. Eng. Guilherme Fernandes dos Santos (EMC/UFG). Após a apresentação, a banca examinadora realizou a arguição do(a) estudante. Posteriormente, de forma reservada, a Banca Examinadora atribuiu a nota final de 9,0 (nove), tendo sido o TCC considerado aprovado.

Proclamados os resultados, os trabalhos foram encerrados e, para constar, lavrou-se a presente ata que segue assinada pelos Membros da Banca Examinadora.

Documento assinado eletronicamente por **Geyverson Teixeira De Paula**, **Professor do Magistério Superior**, em 22/07/2024, às 15:03, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

Documento assinado eletronicamente por **Leonardo Da Cunha Brito**, **Professor do Magistério Superior**, em 22/07/2024, às 15:04, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

Documento assinado eletronicamente por **Guilherme Fernandes Dos Santos**, **Usuário Externo**, em 22/07/2024, às 15:06, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4687509** e o código CRC **23BB0A26**.

Emprego de Inteligência Artificial na Identificação de Parâmetros Eletromagnéticos de Motores Elétricos

João P. A. F. Matos,
Pedro P. C. Vieira,
Escola de Engenharia Elétrica, Mecânica e de Computação Universidade Federal de Goiás
Goiânia, Goiás

***Resumo*— Este trabalho investiga o desenvolvimento de uma aplicação de inteligência artificial para identificar parâmetros eletromagnéticos em motores elétricos. Utilizando o software FEMM, foi criado um conjunto de dados de curvas de fluxo magnético por corrente para diversos materiais ferromagnéticos, que serviu de base para o treinamento de redes neurais. Estas redes foram capazes de encontrar a equação dessas curvas a partir de um número mínimo de pontos, independentemente do tipo de material, implicando em possíveis melhorias na eficiência e desempenho de motores elétricos.**

Palavras-chave: Inteligência Artificial, Parâmetros Eletromagnéticos, Motores Elétricos, Software FEMM, Redes Neurais, Curvas de Fluxo Magnético, Materiais Ferromagnéticos, Ajuste de Curvas, Eficiência de Motores, MATLAB.

***Abstract*—This work investigates the development of an artificial intelligence application to identify electromagnetic parameters in electric motors. Using the FEMM software, a dataset of magnetic flux curves by current for various ferromagnetic materials was created, serving as the basis for training neural networks. These networks were able to determine the equation of these curves from a minimal number of points, regardless of the material type, implying possible improvements in the efficiency and performance of electric motors.**

***Keywords*: Artificial Intelligence, Electromagnetic Parameters, Electric Motors, FEMM Software, Neural Networks, Magnetic Flux Curves, Ferromagnetic Materials, Curve Fitting, Motor Efficiency, MATLAB.**

I. INTRODUÇÃO

Nas últimas décadas, o avanço tecnológico transformou significativamente diversos setores da indústria, incluindo o campo dos motores elétricos. A eficiência e a performance desses motores são essenciais para uma vasta gama de aplicações, desde pequenos dispositivos eletrônicos até grandes maquinários industriais. Uma área emergente que promete revolucionar a forma como entendemos e otimizamos esses motores é a aplicação de inteligência artificial (IA) para a identificação de parâmetros eletromagnéticos.

A identificação desses parâmetros antes da partida do motor é importante visto que os parâmetros do motor podem mudar com o envelhecimento, temperatura de operação e que, em caso de troca de motor, o inversor com essa abordagem, será capaz de identificar o parâmetros de uma grande faixa de motores, permitindo o *auto-tunning* e o auto comissionamento.

II. MODELO DE IDENTIFICAÇÃO DAS CORRENTES E FLUXO MAGNÉTICO

A. Parâmetros Eletromagnéticos

Este trabalho tem como objetivo aproximar a curva de corrente em função do fluxo magnético de um motor elétrico, utilizando leituras de dados esparsos obtidas do motor. Motores elétricos exibem comportamento de saturação magnética não-linear, onde o fluxo magnético não aumenta proporcionalmente à corrente. Modelar essa característica é crucial para o controle e predição de performance do motor.

Diversos métodos para a identificação de parâmetros magnéticos em condições de repouso têm sido amplamente estudados e aplicados tanto a SyRMs quanto a motores de ímã permanente interior (IPMs). Métodos como os de Štumberger et al. [1], Štumberger et al [2] e Peretti et al. [3] aplicam pulsos de tensão ou corrente ao motor e calculam as ligações de fluxo pela integração das tensões induzidas. Contudo, devido à baixa magnitude dos pulsos aplicados, esses métodos são suscetíveis a erros na resistência do estator e na tensão do inversor.

Para superar as limitações identificadas nesses métodos, Hinkkanen et al. [4] propuseram um modelo magnético algébrico que oferece um bom compromisso entre o número de parâmetros e a precisão. Este modelo é baseado em polinômios que descrevem as características de saturação de autoeixo e cruzamento de saturação, além de ser numericamente invertível, o que o torna conveniente para uso em processos de autocomissionamento.

A integração de V_d subtraído de $R_s I_d$, assumindo que $R_s I_d$ seja pequeno, fornece diretamente a ligação de fluxo Φ, conforme a equação abaixo:

$$\psi_d(t) = \int \left[u_d(t) - R_s i_d(t)\right] dt \tag{1}$$

Similarmente, para o eixo q:

$$\psi_q(t) = \int \left[u_q(t) - R_s i_q(t)\right] dt \tag{2}$$

onde ψ_d e ψ_q são as componentes de fluxo, u_d e u_q são as componentes de tensão, e R_s é a resistência do estator.

As correntes I_d e I_q são modeladas como funções não lineares das componentes de fluxo, conforme as equações abaixo:

$$i_d = i_d(\psi_d, \psi_q) \tag{3}$$

$$i_q = i_q(\psi_d, \psi_q) \tag{4}$$

As características de saturação do eixo d podem ser modeladas inicialmente pela seguinte equação polinomial:

$$i_d(\psi_d) = (a_{d0} + a_{d1}|\psi_d| + \cdots + a_{dn}|\psi_d|^n)\psi_d \tag{5}$$

onde $a_{d0} \ldots a_{dn}$ são os coeficientes e n é o maior expoente.

O polinômio (5) pode ser simplificado:

$$i_d(\psi_d) = \left(a_{d0} + a_{dd}|\psi_d|^S\right)\psi_d \tag{6}$$

onde a_{d0} é o inverso da indutância não saturada, S é um expoente positivo determinando a forma das características de saturação, e a_{dd} é um coeficiente não negativo.

Portanto, a indutância pode ser expressa como:

$$L_d(\psi_d) = \frac{\psi_d}{i_d(\psi_d)} = \frac{1}{a_{d0} + a_{dd}|\psi_d|^S} \tag{7}$$

A eficácia do método proposto foi validada experimentalmente no artigo de [4] utilizando um SyRM de 2,2 kW. Os resultados mostraram que o modelo ajustado corresponde muito bem às características de saturação medidas pelo método de rotação constante, validando sua aplicabilidade para autocomissionamento automático de *drivers* de SyRM sem sensores em repouso.

Para garantir que o polinômio (5) ofereça uma aproximação adequada, o coeficiente n deve ser suficientemente elevado, geralmente entre 4 e 7 (Hinkkanen et al., 2017 [4]), como pode ser visto na Figura 1.

A identificação precisa do modelo magnético é crucial para o desempenho de motores de relutância síncrona (SyRMs), especialmente sob condições de parada. Métodos como a aplicação de pulsos de tensão bipolar em sequência curta são utilizados para determinar a saturação e os efeitos de saturação cruzada, integrando as tensões induzidas para calcular as ligações de fluxo do estator. Para realizar a predição dessa curva de corrente por fluxo, o projeto propõe o desenvolvimento de uma rede neural profunda do tipo MLP.

B. Rede neural MLP

A arquitetura MLP é uma classe de redes neurais *feed-forward*, onde os dados fluem em uma única direção, da entrada para a saída. Cada neurônio em uma camada é conectado a todos os neurônios da próxima camada como mostrado na Figura 2. O aprendizado em MLPs é geralmente supervisionado e realizado através do algoritmo de retropropagação, que ajusta os pesos das conexões para minimizar o erro da rede.

Considere uma MLP com uma camada de entrada, uma camada oculta e uma camada de saída. Seja $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ o vetor de entrada e $\mathbf{y} = [y_1, y_2, \ldots, y_m]$ o vetor de saída.

1) Camada Oculta::

$$h_j = \sigma\left(\sum_{i=1}^{n} w_{ij}^{(1)} x_i + b_j^{(1)}\right) \tag{8}$$

Onde $w_{ij}^{(1)}$ é o peso da conexão entre o neurônio i da camada de entrada e o neurônio j da camada oculta, $b_j^{(1)}$ é o viés do neurônio j na camada oculta, e σ é uma função de ativação não-linear, como a função sigmoide ou ReLU.

2) Camada de Saída::

$$y_k = \sigma\left(\sum_{j=1}^{p} w_{jk}^{(2)} h_j + b_k^{(2)}\right) \tag{9}$$

onde $w_{jk}^{(2)}$ é o peso da conexão entre o neurônio j da camada oculta e o neurônio k da camada de saída, $b_k^{(2)}$ é o viés do neurônio k na camada de saída, e σ é a função de ativação apropriada.

C. Algoritmo de Retropropagação

O algoritmo de retropropagação é utilizado para treinar a MLP, ajustando os pesos e vieses para minimizar a função de custo. A função de custo comumente utilizada é o erro quadrático médio (MSE):

$$E = \frac{1}{2}\sum_{k=1}^{m}(y_k - \hat{y}_k)^2 \tag{10}$$

Sendo y o valor real e $\hat{y}_k$ o valor predito pela rede neural. A atualização dos pesos é feita através do gradiente descendente:

1) Cálculo do Gradiente::

$$\frac{\partial E}{\partial w_{jk}^{(2)}} = \delta_k h_j \tag{11}$$

Onde $\delta_k = (y_k - \hat{y}_k)\sigma'(\text{net}_k)$,
e $\text{net}_k = \sum_j w_{jk}^{(2)} h_j + b_k^{(2)}$.
Para a camada oculta:

$$\frac{\partial E}{\partial w_{ij}^{(1)}} = \delta_j x_i \tag{12}$$

onde $\delta_j = \left(\sum_k \delta_k w_{jk}^{(2)}\right)\sigma'(\text{net}_j)$.

2) Atualização dos Pesos::

$$w_{ij}^{(t+1)} \leftarrow w_{ij}^{(t)} - \eta\frac{\partial E}{\partial w_{ij}^{(t)}} \tag{13}$$

$$w_{jk}^{(t+2)} \leftarrow w_{jk}^{(t+1)} - \eta\frac{\partial E}{\partial w_{jk}^{(t+1)}} \tag{14}$$

onde η é a taxa de aprendizado.

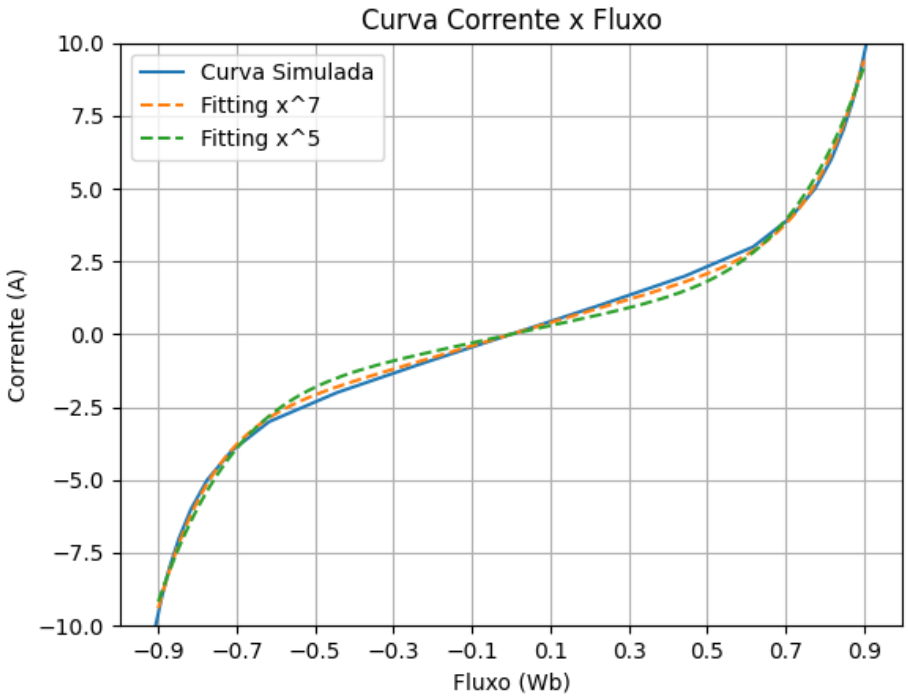

Figura 1: Comparação entre os *fittings*: x^7 x x^5

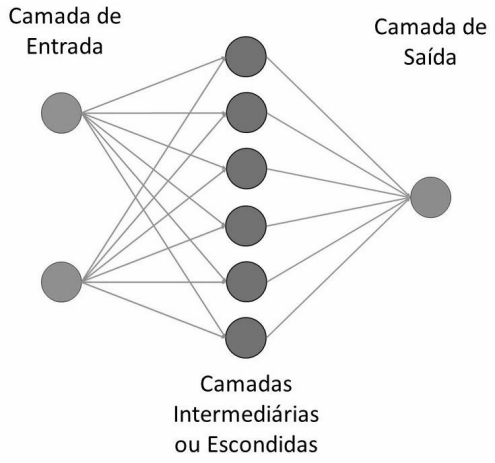

Figura 2: MLP genérica

D. Utilização em Ajuste de Curvas com Dados Esparsos

Os MLPs são especialmente úteis para ajuste de curvas (*curve fitting*) em situações onde os dados disponíveis são esparsos. A capacidade das redes neurais de modelar relações complexas e não-lineares permite que elas ajustem bem os dados limitados, desde que a estrutura da rede e o processo de treinamento sejam adequados.

E. Algoritmos de Treinamento de Redes Neurais

1) **Levenberg-Marquardt (trainlm)**
   - **Uso:** Adequado para redes de pequeno a médio porte.
   - **Características:** Combina a velocidade do método de Gauss-Newton com a robustez da descida de gradiente.
   - **Vantagens:** Convergência muito rápida para muitos problemas.
   - **Desvantagens:** Requer muita memória, tornando-o inadequado para redes ou conjuntos de dados grandes.
2) **Regularização Bayesiana (trainbr)**
   - **Uso:** Adequado para problemas onde o *overfitting* é uma preocupação.
   - **Características:** Modifica o algoritmo de Levenberg-Marquardt incorporando a regularização bayesiana para melhorar a generalização.
   - **Vantagens:** Reduz o *overfitting*, mesmo para pequenos conjuntos de dados.
   - **Desvantagens:** Mais lento que o algoritmo de Levenberg-Marquardt padrão.
3) **Gradiente Conjugado Escalonado (trainscg)**
   - **Uso:** Adequado para redes e conjuntos de dados grandes.
   - **Características:** Uma variante do método de gradiente conjugado que escala bem com o tamanho do problema.
   - **Vantagens:** Requer menos memória e é mais rápido para grandes conjuntos de dados.
   - **Desvantagens:** Convergência mais lenta em comparação com Levenberg-Marquardt para pequenos

conjuntos de dados.

4) **Quasi-Newton BFGS (trainbfg)**
   - **Uso:** Adequado para redes de tamanho médio.
   - **Características:** Utiliza uma aproximação da matriz Hessiana para encontrar o ótimo.
   - **Vantagens:** Geralmente convergência mais rápida que a descida de gradiente.
   - **Desvantagens:** Requer mais memória e pode ser mais lento que Levenberg-Marquardt.
5) **Retropropagação Resiliente (trainrp)**
   - **Uso:** Adequado para redes com grandes variações nas magnitudes dos gradientes.
   - **Características:** Considera apenas o sinal do gradiente, não sua magnitude, o que o torna robusto contra grandes variações de gradientes.
   - **Vantagens:** Eficaz para redes rasas e pode lidar bem com gradientes ruidosos.
   - **Desvantagens:** Não é adequado para redes muito profundas ou grandes conjuntos de dados.
6) **Fletcher-Powell Conjugate Gradient (traincgf)**
   - **Uso:** Adequado para redes e conjuntos de dados grandes.
   - **Características:** Uma variante do método de gradiente conjugado que ajusta a direção de busca usando a atualização de Fletcher-Reeves.
   - **Vantagens:** Melhores propriedades de convergência para alguns tipos de problemas.
   - **Desvantagens:** Semelhante a outros métodos de gradiente conjugado, pode ser mais lento que o Levenberg-Marquardt para redes pequenas.
7) **Gradient Descent (traingd)**
   - **Uso:** Adequado para redes rasas e pequenos conjuntos de dados.
   - **Características:** O algoritmo de otimização mais básico que atualiza os pesos pela direção negativa do gradiente da função de perda.
   - **Vantagens:** Simples e fácil de implementar.
   - **Desvantagens:** Convergência muito lenta e pode ficar preso em mínimos locais.

Além destes algoritmos existem vários outro mas para este trabalho serão utilizados apenas os algoritmos citados acima.

III. METODOLOGIA

A. Seleção e Preparação dos Materiais

Para este projeto, foi essencial selecionar uma variedade de materiais magnéticos comumente utilizados em motores elétricos, além de uma seleção de materiais amplamente documentados disponíveis na internet [6]. A diversidade no conjunto de dados visava garantir um treinamento robusto da rede neural, permitindo uma modelagem precisa das curvas de fluxo magnético por corrente. A escolha dos materiais foi baseada em suas propriedades magnéticas e na disponibilidade de dados documentados. Inicialmente foram selecionados 50 materiais com característica magnética, mas eventualmente foram filtrados, de forma a priorizar aqueles mais amplamente utilizados em motores elétricos, de forma a garantir uma maior relevância prática aos resultados.

B. Simulações

Para realizar foi utilizado o software FEMM em conjunto com a linguagem de programação *Python*. As simulações consistiam em desenhar a geometria do motor no FEMM, e coletar os dados de fluxo magnético, para determinadas correntes em suas bobinas. As simulações foram realizadas de forma automatizadas com o uso da biblioteca pyFEMM, que consiste em uma interface em *Python* para o FEMM. Inicialmente foram feitas simulações considerando apenas um transformador, por esse possuir uma geometria mais simples, de forma a se construir um módulo em *Python* capaz de realizar toda a simulação de forma automatizada, essa consistindo na coleta dos fluxos magnético, quando aplicada uma certa corrente nas bobinas. Simulando a corrente de -10 a 10A, variando de 1 em 1A. Depois, foi modificada a geometria para um modelo de uma SRM(Motor de Relutância Síncrona) 4/6 disponível no Laboratório de Ensaios de Pequenos Motores (LEnP-Mot) da EMC/UFG, como demonstrado no modelo da Figura 3, e realizadas todas as simulações de forma a montar um *dataset* completo com as curvas de fluxo por corrente para cada um dos materiais coletados.

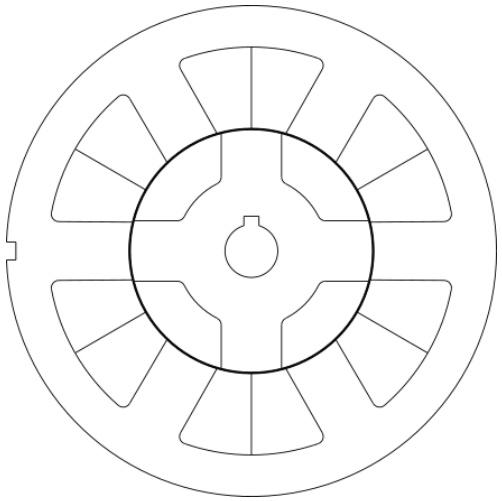

Figura 3: Geometria do motor no FEMM

C. Métodos

Foram utilizados três métodos distintos para a geração da rede neural e o tratamento dos dados. Esses métodos foram desenvolvidos ao longo do projeto e estão descritos a seguir:

1) Método 1: Os dados foram aproximados utilizando o método dos mínimos quadrados múltiplos e empregados no treinamento da rede neural, com pontos aleatórios servindo como entrada e os coeficientes da curva representada pela equação (6), a e b, como saída.

2) Método 2: Um conjunto de dados sintéticos foi gerado utilizando a equação algébrica $ax^7 + bx$ para a criação dos dados de entrada e saída com a e b dentro das faixas observadas nos dados reais, que foi utilizado para o treinamento

da rede neural. Tanto as entradas quanto as saídas a e k (sendo k definido como a razão entre a e b) eram geradas aleatoriamente dentro de uma faixa definida. Enquanto isso, os dados aproximados foram reservados exclusivamente para teste.

3) Método 3: Neste método, foi utilizada a criação de dados sintéticos semelhante ao Método 2, porém, desta vez, foram realizadas todas as permutações possíveis de dos valores de corrente para cada a e b possível, dentro das faixas observadas nos dados reais. Após isso, foi realizada uma análise estatística das inferências da rede neural, empregando um método de otimização para encontrar os valores de entrada que resultavam nas melhores saídas.

D. Preparação dos Dados

Após a coleta das curvas de fluxo por corrente, foi realizado um ajuste de curva (*curve fitting*) utilizando o MATLAB para obter as equações aproximadas das curvas. A característica da curva proposta pelo artigo de Hinkkanen et al. [4] foi uma curva utilizando apenas dois coeficientes, a e b, pois eles são suficientes para capturar as características não lineares dos dados. No artigo, foram propostas curvas com dimensionalidade variando entre $y = ax^4 + bx$ até $y = ax^7 + bx$. Foram testados diferentes dimensões, e optamos por utilizar, para nossos experimentos, a curva $y = ax^7 + bx$ para os ajustes de curva (*curve fittings*), pois entre todos as dimensões testadas, ela teve uma melhor aproximação para os dados simulados.

Os dados coletados, pontos da curva corrente x fluxo, foram divididos em conjuntos de teste os quais foram filtrados a fim de que os dados fossem o mais próximo possível da utilização real com materiais ferromagnéticos de motores que são muito utilizados na indústria, retirando-se assim diversos *outliers*.

No método 1 os dados coletados não se mostraram suficientes para o treinamento da rede, resultando em *overfitting* da rede em poucas épocas, sem que a rede generalizasse de forma satisfatória.

Portanto utilizamos o método 2, nele foram gerados dados sintéticos que simulavam o comportamento dos materiais coletados. Dessa forma, foi observado que havia uma certa relação de proporcionalidade entre os valores de a e b. Com isso, foi feita uma análise de regressão com todos os pontos de todos os materiais e encontrado um valor de coeficiente angular que seria o valor de correlação entre os valores de a e b, que variava de 3 a 14 dependendo do material. Nomeamos este coeficiente de k, definido como $k = a/b$.

Ao redor deste valor e dentro de um escopo de a, também definido pelo nosso conjunto de dados simulados, variando de 1 a 50, foi feita a geração dos dados artificiais para o treinamento da rede. Isso permitiu que, mesmo com um número pequeno de amostras, fosse possível treinar a rede com mais de 5 mil dados de entrada.

E. Desenvolvimento da Rede Neural

Optou-se pelo uso de uma rede neural profunda a fim de que pudéssemos aproximar as curvas com uma quantidade limitada ou ate mesmo esparsa de pontos a fim de que a rede possa ser geral o suficiente a para que consiga aproximar a curva escolhida sem necessitar de muitas medidas de pontos, além disso foi colocado uma grande ênfase em modelos pequenos para que a rede pudesse ser facilmente embarcada em um microcontrolador arbitrário, estes 2 fatores foram grandes direcionadores e também de certa forma limitadores no desenvolvimento do projeto.

A rede neural escolhida para este projeto foi uma Multilayer Perceptron (MLP), conhecida por sua capacidade de modelar relações não-lineares e também por seu relativo baixo custo computacional em relação a outros modelos utilizados, aumentando então a gama de microcontroladores que poderiam ser utilizados. Para se definir a arquitetura da MLP foram feitos diversos testes, modificando seu número de camadas ocultas e características e algoritmos comumente utilizados para treino, de forma a se encontrar a que oferecia uma melhor resposta.

Para o treinamento inicial da rede neural, utilizamos a linguagem de programação *Python* devido à sua flexibilidade e à ampla gama de bibliotecas disponíveis. No entanto, conforme o projeto avançou, tornou-se necessário adotar algoritmos mais robustos e estruturar testes automatizados de forma mais eficiente. Por isso, optamos por utilizar a ferramenta *Neural Fitting* do MATLAB, que permitiu a automatização dos treinamentos de diferentes arquiteturas e a aplicação de diversos algoritmos de treinamento comumente usados para ajuste de curvas. Além disso, a vasta documentação e a comunidade ativa em fóruns online do MATLAB forneceram um suporte valioso durante o desenvolvimento do projeto.

Para garantir a eficácia do treinamento, realizamos uma série de testes com diferentes algoritmos de treinamento, considerando suas vantagens e desvantagens específicas. Também se variou diversos parâmetros, como o número de épocas e o número de camadas ocultas na rede neural, a fim de identificar a configuração ótima para o problema em questão. Como as redes MLP (*Multilayer Perceptron*) requerem um tempo de treinamento relativamente curto, conseguimos simular diversas redes em um período de tempo reduzido, ajustando esses parâmetros de maneira eficiente.

F. Avaliação do Modelo

O desempenho da rede neural foi avaliado utilizando gráficos de predição versus valores reais, permitindo a visualização da precisão das previsões da rede.

Os critérios de avaliação incluíram métricas como o erro quadrático médio (MSE) e o coeficiente de determinação (R^2), que forneceram uma medida quantitativa da precisão do modelo para diferentes algoritmos e conjuntos de dados.

IV. Resultados

Esta seção apresenta os resultados obtidos durante o desenvolvimento do projeto, destacando a importância desses resultados para os objetivos estabelecidos. A análise dos dados inclui simulações de motores elétricos e o desenvolvimento

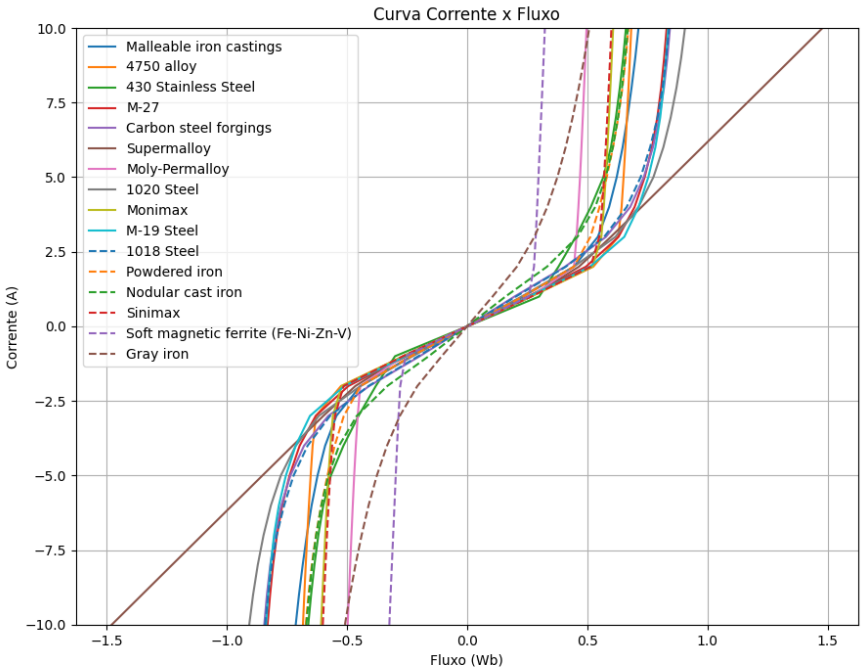

Figura 4: Curvas de fluxo por corrente para diferentes materiais

de um modelo de rede neural para identificar parâmetros eletromagnéticos.

Primeiramente foram obtidas as curvas de fluxo magnético por corrente por meio das simulações realizadas no software FEMM utilizando o modelo do motor elétrico apresentado na Figura 3. As curvas obtidas para diferentes materiais são apresentadas na Figura 4. Essas curvas mostram as características magnéticas dos materiais simulados.

Para reduzir os *outliers* e manter a análise da rede dentro do escopo comercial, focamos nos materiais mais comumente utilizados em motores elétricos, conforme indicado no catálogo AkSteel[8] e no livro SMAG Handbook[7]. Compilamos uma lista dos materiais mencionados nesses recursos e que também estavam presentes no conjunto de dados. A maior parte dos materiais utilizados pode ser classificada em aços de baixo carbono e aços elétricos.

A. Aços de Baixo Carbono

Os aços de baixo carbono são caracterizados pelo seu baixo teor de carbono, o que os torna relativamente macios e dúcteis. Essas características são ideais para aplicações que exigem alta permeabilidade magnética e baixa coercividade. Entre os aços de baixo carbono comumente utilizados estão:

- 1006
- 1010
- 1018
- 1020
- 1117

Esses aços são frequentemente utilizados em núcleos magnéticos de motores elétricos devido à sua capacidade de suportar altas densidades de fluxo magnético com perdas mínimas de energia.

B. Aços Elétricos

Os aços elétricos são projetados especificamente para aplicações magnéticas. Eles possuem diferentes teores de silício, que aumentam a resistividade elétrica e reduzem as perdas por correntes parasitas. As classes de aços elétricos mais comumente utilizadas incluem:

- M-15
- M-19
- M-22
- M-27
- M-36
- M-43
- M-45

Esses aços são amplamente utilizados em núcleos laminados de motores elétricos e transformadores. As classes M-19, M-22 e M-27 são especialmente populares devido ao seu equilíbrio entre custo, desempenho e fabricabilidade.

As curvas dos materiais mencionados foram filtradas, mantendo-se apenas as curvas dos materiais mais relevantes. As curvas dos materiais selecionados estão apresentadas na Figura 5.

Após as simulações foi realizado a interpolação das curvas, obtendo-se os coeficientes a e b, para uma curva equação do tipo $y = ax^7 + bx$, de forma que se aproximem das

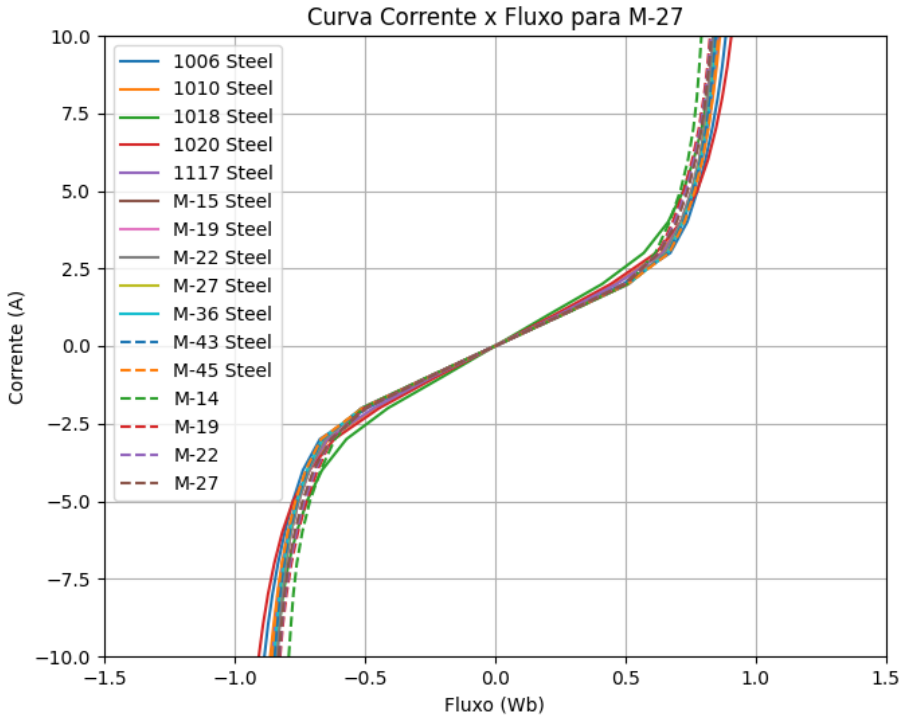

Figura 5: Curvas de fluxo por corrente selecionados

curvas simuladas. A comparação entre a curva interpolada e a curva simulada do material 1010 Steel está apresentada na Figura 6, todas as interpolações das curvas dos materiais apresentados estão presentes no Apêncice A. Essas curvas são importantes para poder realizar o treinamento da rede, e avaliar sua eficiência.

Inicialmente utilizou-se estes dados para treinamento da rede neural, porem por conta da quantidade pequena de curvas utilizou-se um *dataset* sintético das curvas de $y = ax^7 + bx$, as curvas foram geradas com valores aleatórios de a e b, e os respectivos valores de fluxo para um vetor de correntes com valores de 1 a 10.

A ideia inicial para o projeto foi minimizar a quantidade de pontos para a inferência do modelo, diminuindo assim a complexidade da rede necessária e a quantidade de testes para uma estimativa correta, dessa forma iniciamos os testes com 3 pontos (x1,y1,x2,y2,x3,y3) como entrada na rede, pois eram o mínimo valor de pontos que conseguiam render uma aproximação de uma curva e como saída valores de a e b.

C. Método 1

Foram realizados diversos testes utilizando 2 pontos, selecionados de forma aleatória, variando o número de camadas ocultas, entretanto a rede não convergia, tendo um MSE extremamente alto na casa de 10^3. Para tentar melhorar esse resultado, tentou-se treinar utilizando - se mais pontos, mas ainda assim o melhor MSE alcançado não foi satisfatório. Primeiramente acreditava -se que poderia ser um problema com a normalização dos dados, entretanto ao se testar diferentes métodos de normalização, não houve melhora notável na rede. Portanto chegamos a conclusão que o problema para se treinar a rede se dava por conta do pouco número de dados. Havendo apenas 50 materiais, com 10 pontos cada, ou seja, um total de 500 pontos para teste e treino, o que se provou insuficiente para a convergência da rede. O gráfico da função de perda pode ser observado na Figura 7, e a diferença entre a curva real e a predita na Figura 8

D. Método 2

Após a falha nos testes com dados reais optou-se por utilizar um algoritmo de criação de dados sintéticos, para aumentar a variabilidade dos dados e também a sua quantidade, esperando que este método se adaptasse bem à distribuição dos dados reais. Além disso optou-se por utilizar o Matlab com o *addon Neural Network Toolbox* para realizar o treino, as inferências e estudo dos dados. O Matlab permite que possamos fazer uma quantidade elevada de testes e ter todas os gráficos de forma bem ágil e otimizada, além disso o Matlab também permite a escolha de diversos algoritmos de treinamento amplamente utilizados na interpolação de curvas, estes algoritmos fazem com que o treinamento seja mais eficiente para tarefas especificas, a taxa de aprendizado de cada um dos treinos é definida de forma dinâmica por cada um dos algoritmos utilizados.

Para este experimento, desenvolvemos um algoritmo que varia os seguintes parâmetros para treinar uma rede neural no MATLAB:

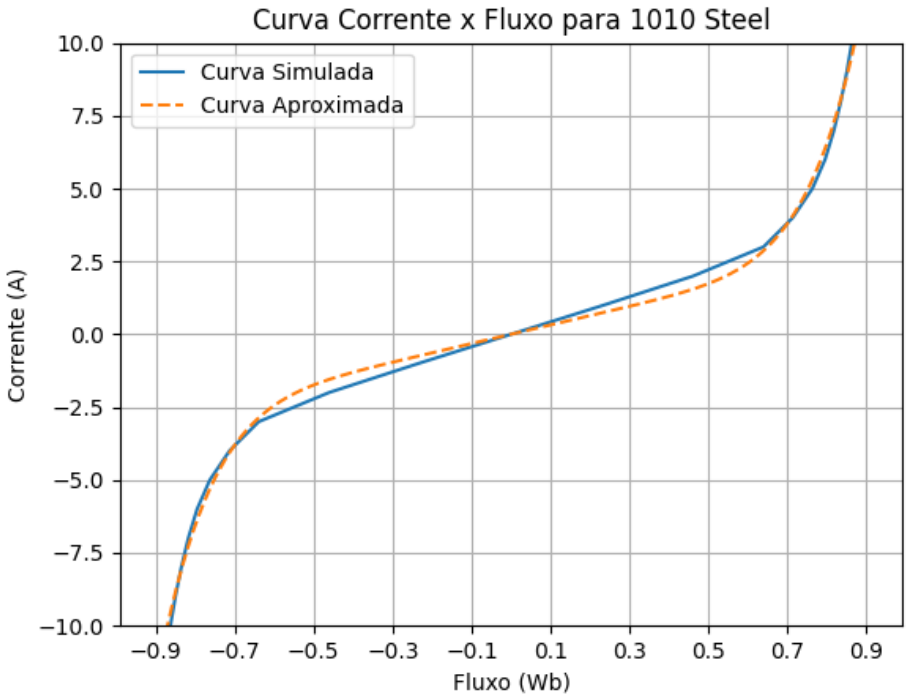

Figura 6: Curva Simulada X Curva Aproximada para 1010 Steel

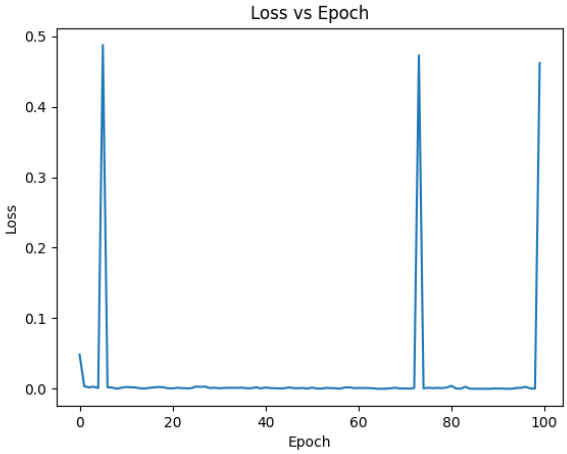

Figura 7: Função de perda durante o treinamento com dados aproximados

- Épocas: de 5 a 50, incrementando de 5 em 5.
- Neurônio: de 1 a 50, incrementando de 1 em 1 ate o 5 neurônio e de 5 em 5 após isso.
- Algoritmos de treinamento: `trainbfg`, `trainscg`, `traingdm`, `traincgf` ou `trainrp`.

Para cada combinação de parâmetros, o algoritmo realiza o seguinte processo:

1) Configura a rede neural com o número especificado de camadas ocultas.

Épocas	**Neurônios**	**Algoritmos**
5	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
10	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
15	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
20	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
25	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
30	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
35	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
40	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
45	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `trainrp`
50	1 a 50	`trainbfg`, `trainscg`, `traingdm`, `traincgf`, `traincgf`

Tabela I: Parâmetros dos testes realizados

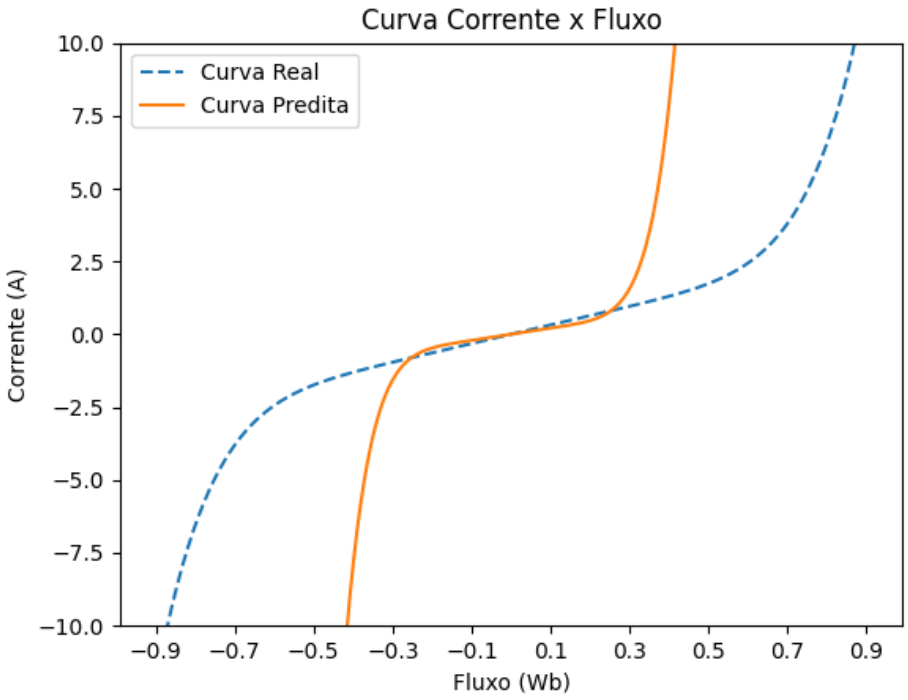

Figura 8: Curva real X Curva Predita para treinamento com dados aproximados

Algoritmo	Neurônios	Épocas	MSE	R
traingdm	2	40	26.48	0.80
traincgf	1	5	26.96	0.79
trainrp	2	35	26.97	0.79
trainbfg	4	10	27.12	0.80
trainscg	2	30	27.27	0.79

Tabela II: Melhores performances dos testes realizados

2) Treina a rede neural utilizando o número especificado de épocas.
3) Utiliza um dos algoritmos de treinamento: `trainbfg`, `trainscg`, `traingdm`, `traincgf` ou `trainrp`.
4) Avalia o desempenho da rede neural e registra os resultados em um csv.

Os hiperparâmetros para os 5 algoritmos com maior performance estão presentes na Tabela II, esses algoritmos são apenas alguns de todos os que melhor performaram do teste elucidado pela Tabela I.

Analisando os dados, podemos observar que o algoritmo 'traingdm' teve o melhor desempenho com 2 camadas ocultas e 40 épocas. No entanto, seu MSE ainda é consideravelmente alto. Isso fica evidente ao examinar o gráfico da função de perda, Figura 9, o gráfico de *fitting*, Figura 10, e os histogramas de inferência, Figura 11, que mostram discrepâncias significativas nas previsões.

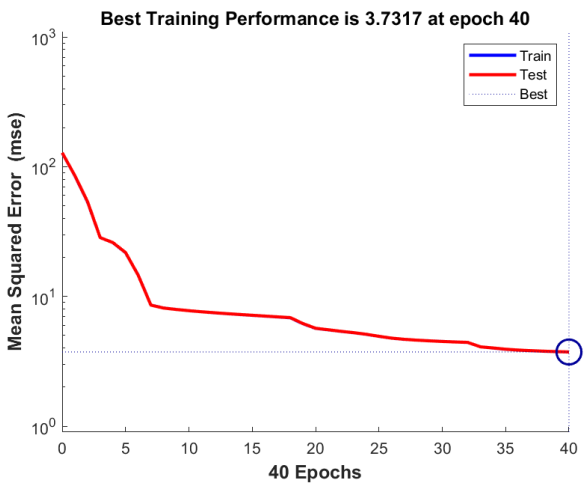

Figura 9: Função de perda do treinamento para traingdm 40 épocas

Podemos observar pelo gráfico de *fitting* que estes dados não ficaram satisfatórios, a rede aparenta estar pegando o valor médio das curvas e não alterando a medida que os valores de entrada mudam, por isso a característica paralela ao eixo X da Figura 10.

Testamos então esta rede para cada um dos materiais mais utilizados na industria

E. Método 3

Observamos na Tabela III que havia inconsistências nos resultados, sugerindo problemas na concepção do algoritmo

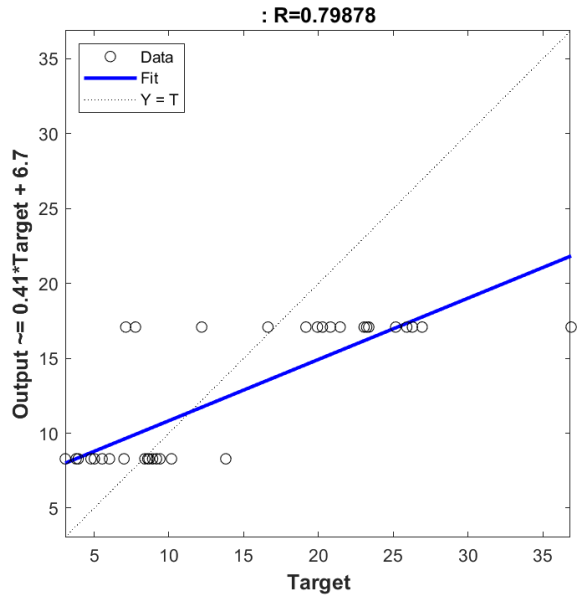

Figura 10: Dados ajustados pela rede traingdm 40 épocas

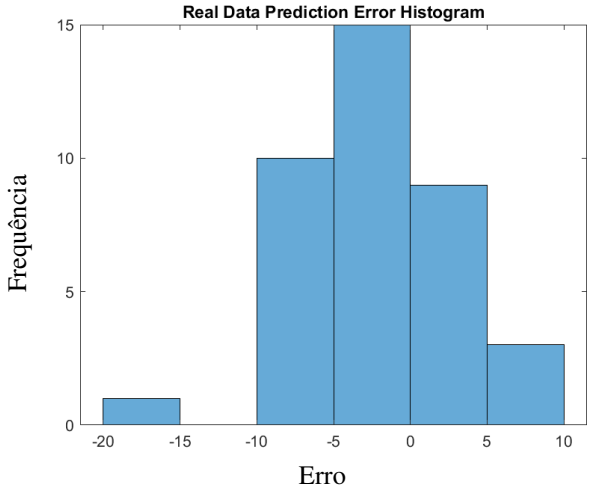

Figura 11: Histograma de erro da rede traingdm 40 épocas

Material	a_{fit}	k_{fit}	a_{nn}	k_{nn}
1006 Steel	16.62	5.534	17.085	5.534
1010 Steel	19.16	6.025	17.085	5.534
1018 Steel	20.26	4.787	17.085	5.534
1020 Steel	12.19	3.070	17.085	5.534
1117 Steel	20.8	6.996	17.085	5.534
M-15 Steel	25.16	10.17	17.085	5.534
M-27 Steel	23.22	9.163	17.085	5.534
M-36 Steel	23.05	9.168	17.085	5.534
M-43 Steel	23.36	9.396	17.085	5.534
M-45 Steel	21.45	8.388	17.085	5.534
M-14	36.89	13.80	17.085	5.534
M-19	26.93	8.581	17.085	5.534
M-22	26.28	8.667	17.085	5.534
M-27	25.9	8.909	17.085	5.534

Tabela III: Dados Aproximados e Dados Preditos com o traingdm 40 épocas

Algoritmo	Camadas Ocultas	Épocas	MSE	R
trainscg	15	40	10.617	0.937
trainbfg	5	30	10.837	0.929
trainscg	5	40	11.604	0.922
trainscg	15	20	12.298	0.921
trainbfg	15	40	12.307	0.923

Tabela IV: melhores performances com o *dataset* alterado

ou na ideia subjacente. Portanto, revisamos a criação do *dataset* sintético e identificamos que nosso código gerava valores aleatórios para a e b, realizando apenas uma amostragem aleatória de pontos com esses valores. Isso dificultava a capacidade da MLP (Multi-Layer Perceptron) de capturar eficientemente as características dos dados. Para corrigir esse problema, modificamos a abordagem para que a rede recebesse todas as combinações possíveis dos elementos do vetor corrente tomados 4 de cada vez (variando x1,x2,x3,x4), resultando em 210 entradas possíveis, para cada combinação de a e b, com estes variando de 0,1 em 0,1 dentro do intervalo de 15 a 30 para a e 1 a 7 para b, resultando em 10.500 combinações de a e b. Essa alteração resultou em um *dataset* com 2.205.000 linhas. Sendo a saída a e k, sendo $k = a/b$

Com o *dataset* corrigido, realizamos novos testes e observamos uma melhoria significativa na performance da MLP. A rede neural conseguiu capturar de forma mais eficiente as características dos dados, resultando em uma precisão consideravelmente maior. Esse ajuste foi crucial para o avanço da pesquisa, demonstrando a importância de um *dataset* bem estruturado para o treinamento eficaz de modelos de aprendizado de máquina.

Com isso novos testes foram realizados nas mesmas especificações anteriores com o algoritmo que faz diversos treinamentos com isso obtivemos os 5 algoritmos que melhor performaram com esta nova configuração de dados.

Analisando os dados da Tabela IV, observamos que o algoritmo 'trainscg' apresentou o melhor desempenho na captura das características dos dados, utilizando 15 camadas ocultas e 40 épocas. Notamos que seu MSE é consideravelmente mais baixo em comparação aos dados da Tabela I, o que indica uma performance superior do algoritmo na identificação das características dos dados. Além disso, a métrica R no Gráfico 13 também apresenta uma melhoria significativa.

Ao observar os dados de treinamento e teste com as métricas descritas nos dados reais do algoritmo 'trainscg', verifica-se que a rede esta convergindo de forma consideravelmente melhor que a rede treinada nos testes anteriores, a linha de *fitting* aparenta esta pegando as características dos dados de forma muito precisa, mas ainda tende a errar com valores que distam consideravelmente do valor correto.

Na tabela V temos os as predições para os parâmetros a e k nomeadas a_{nn} e k_{nn} respectivamente e também os seus respectivos valores aproximados, os valores demonstrados não parecem muito longe dos reais mas podemos ver que ao aleatorizar os dados de entrada na rede em alguns momentos os valores preditos divergem muito dos valores aproximados.

Para aspecto de visualização colocamos a Tabela VI com dados reais e de predição de apenas um material, pode-se

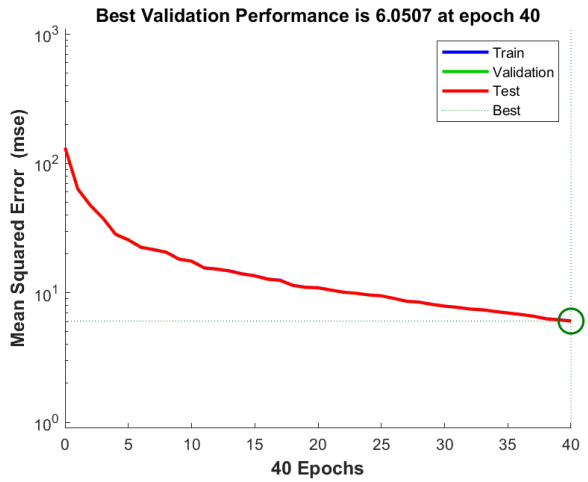

Figura 12: Função de perda do treinamento para trainscg 40 épocas

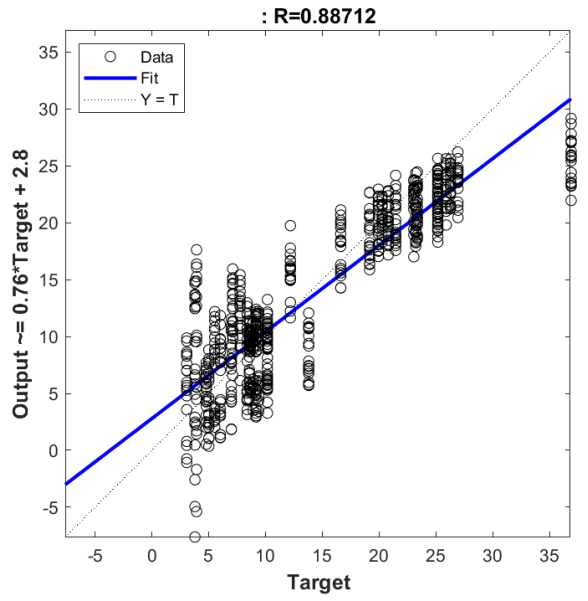

Figura 13: Dados ajustados pela rede trainscg 40 épocas

perceber que em dados momentos as predições divergem bastante do valor aproximado, essa variação evidenciada no gráfico da Figura 13, no qual os valores variarem bastante no eixo y apesar da tendencia geral dos dados tenderem a seguir os valores objetivos. Para mitigar este problema utilizou-se uma analise estatística dos pontos que tendem a gerar melhores resultados (menor MSE) para esta análise utilizou-se uma abordagem baseada na regressão linear múltipla para determinar quais combinações de pontos x minimizam os erros absolutos das predições de a e k.

Para esta analise o método utilizado para encontrar os valores ideais de $x1$, $x2$, $x3$ e $x4$ foi o algoritmo de otimização L-BFGS-B. Este algoritmo busca minimizar o erro total, calculado como a soma dos erros absolutos entre

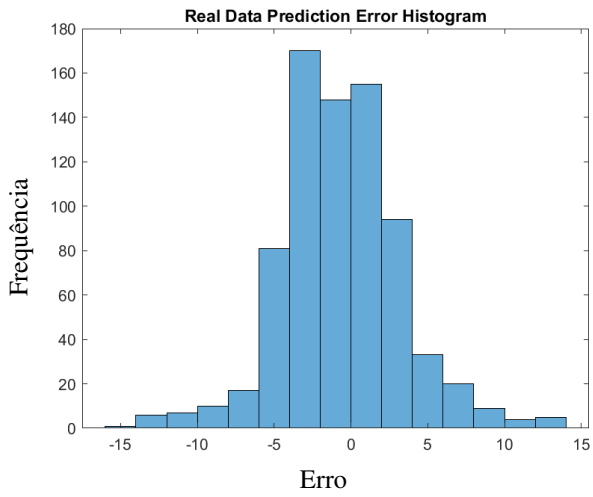

Figura 14: Histograma de erro da rede trainscg 40 épocas

$Materiais$	a_{fit}	k_{fit}	a_{nn}	k_{nn}
1006 Steel	16.62	5.534	18.294	8.863
1006 Steel	16.62	5.534	18.463	8.101
1010 Steel	19.16	6.025	19.347	7.816
1010 Steel	19.16	6.025	19.519	9.794
1018 Steel	20.26	4.787	21.984	6.485
1018 Steel	20.26	4.787	18.687	6.799
1020 Steel	12.19	3.070	12.925	1.645
1020 Steel	12.19	3.070	12.975	1.705
1117 Steel	20.8	6.996	20.172	5.177
1117 Steel	20.8	6.996	19.738	6.346
M-15 Steel	25.16	10.17	24.227	9.746
M-15 Steel	25.16	10.17	22.247	7.491

Tabela V: Dados Aproximados e Dados Preditos com o trainscg 40 épocas

os valores previstos (a_{nn} e k_{nn}) e os valores aproximados (a_{fit} e k_{fit}). Para isso, foi definida uma função objetivo baseada numa combinação linear das variáveis $y1, y2, y3, y4$, com uma suposição inicial de valores $(1, 3, 5, 10)$ e limites estabelecidos para garantir que cada variável permanecesse entre 1 e 10. O processo iterativo de ajuste resultou nos valores otimizados de $x1 = 2.00$, $x2 = 6.00$, $x3 = 8.00$ e $x4 = 10.00$, os quais minimizam o erro entre as previsões e os valores reais do conjunto de dados fornecido.

Com esta analise feita, foi feito o *fitting* dos dados novamente, desta vez utilizando os valores ótimos de $x1 = 2.00$, $x2 = 6.00$, $x3 = 8.00$ e $x4 = 10.00$ e obtemos o seu gráfico de *fitting* na Figura 15 e o histograma de erro na Figura 16.

Pode-se observar que, com os valores ótimos para $x1$, $x2$, $x3$ e $x4$, obtivemos uma curva que se adequa significativamente melhor aos valores objetivos e às características dos dados e do sistema. Além disso, verificamos que o erro, conforme mostrado no histograma da Figura 16, está consideravelmente menor. Com esses valores, agora é possível

I_1	I_2	I_3	I_4	a_{fit}	k_{fit}	a_{rnn}	k_{rnn}
1	2	5	9	16.62	5.5345	14.818	3.5066
5	6	7	10	16.62	5.5345	19.373	5.5884
1	6	8	10	16.62	5.5345	19.069	5.0191
2	3	4	5	16.62	5.5345	16.097	6.5617
1	6	8	10	16.62	5.5345	19.069	5.0191
4	6	8	10	16.62	5.5345	19.606	6.6687
1	7	9	10	16.62	5.5345	19.438	4.625
3	6	8	9	16.62	5.5345	17.639	7.0324
2	3	8	9	16.62	5.5345	16.572	5.7308
4	5	6	8	16.62	5.5345	17.533	5.8632
2	5	6	9	16.62	5.5345	17.359	6.3338
2	3	9	10	16.62	5.5345	17.235	6.1075
2	7	8	9	16.62	5.5345	18.242	5.5596
4	7	8	10	16.62	5.5345	20.448	6.6062
1	6	7	10	16.62	5.5345	18.552	4.933
2	3	5	10	16.62	5.5345	16.242	6.6025
3	4	6	10	16.62	5.5345	16.744	7.3438
2	3	7	9	16.62	5.5345	16.673	5.9291
3	5	8	10	16.62	5.5345	17.956	7.698
2	3	4	5	16.62	5.5345	16.097	6.5617

Tabela VI: Tabela de Dados Aproximados e Dados Preditos para o material 1006 Steel com a rede trainscg

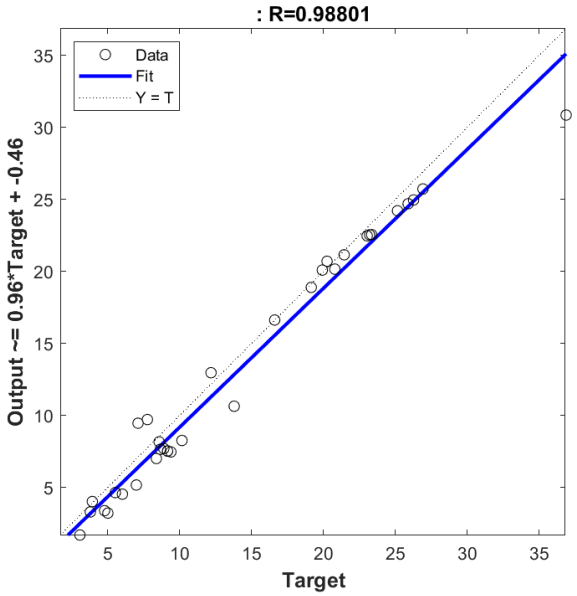

Figura 15: Dados ajustados pela rede trainscg utilizando entrada otimizada

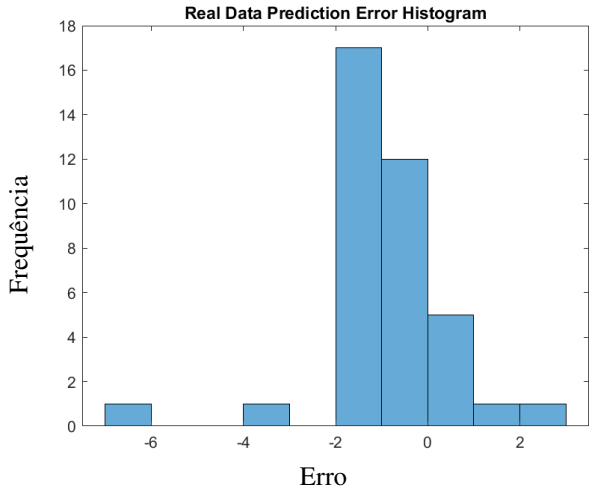

Figura 16: Histograma de erro trainscg utilizando entrada otimizada

obter os dados preditos e reais, conforme apresentado na Tabela VII, e, finalmente, compará-los.

Observa-se nestes dados uma melhora significativa nas predições em comparação com a Tabela V, reforçando ainda mais a importância da qualidade dos dados alimentados para a predição. Neste caso, a rede tende a convergir melhor devido a vários fatores, sendo o mais provável a maior quantidade de dados com valores de $x1 = 2.00$, $x2 = 6.00$, $x3 = 8.00$ e $x4 = 10.00$. Esses valores apresentam uma abrangência da curva suficiente para uma predição de alta qualidade.

Além disso, a análise dos resíduos das predições mostra uma distribuição mais homogênea e com menor variância, as Figuras 17 e 18 apresentam os gráficos dos resíduos calculados para este conjunto de dados antes da analise de pontos ótimos para a rede e as Figuras 19 e 20 apresentam os gráficos dos resíduos calculados para o conjunto de dados com os pontos ótimos. Os resíduos são as diferenças entre os valores observados (reais) e os valores preditos pelo modelo. Eles representam o erro de predição do modelo e são fundamentais para avaliar a precisão e a qualidade

Material	a_{fit}	k_{fit}	a_{nn}	k_{nn}
1006 Steel	16.620	5.534	16.636	4.651
1010 Steel	19.160	6.025	18.902	4.542
1018 Steel	20.260	4.787	20.716	3.395
1020 Steel	12.190	3.070	12.975	1.705
1017 Steel	20.800	6.996	20.172	5.176
M-15 Steel	25.160	10.170	24.227	8.270
M-19 Steel	25.160	10.170	24.227	8.270
M-22 Steel	25.160	10.170	24.227	8.270
M-27 Steel	23.220	9.163	22.554	7.544
M-36 Steel	23.050	9.168	22.484	7.528
M-43 Steel	23.360	9.396	22.567	7.476
M-45 Steel	21.450	8.388	21.163	7.017
M-14	19.930	5.015	20.104	3.227
M-19	26.930	8.581	25.738	8.177
M-22	26.280	8.667	24.976	7.657
M-27	25.900	8.909	24.709	7.708

Tabela VII: Dados Aproximados e Dados Preditos pela rede trainscg utilizando entrada otimizada

das predições. Matematicamente, o resíduo e_i para uma observação i pode ser definido como $e_i = y_i - \hat{y}_i$ onde y_i é o valor observado (real) para a i-ésima observação e $\hat{y}_i$ é o valor predito pelo modelo para a i-ésima observação.

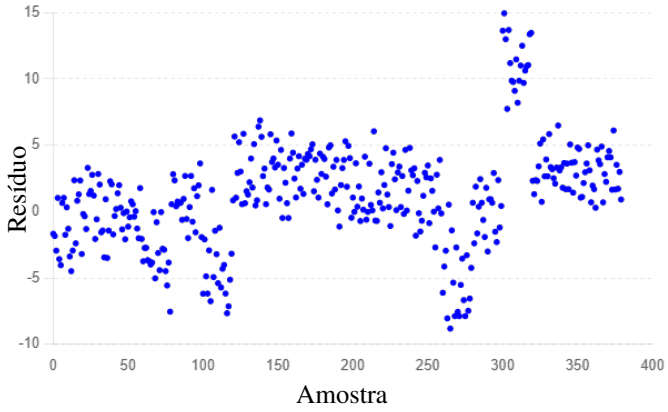

Figura 17: Resíduos de predição a sem entrada otimizada

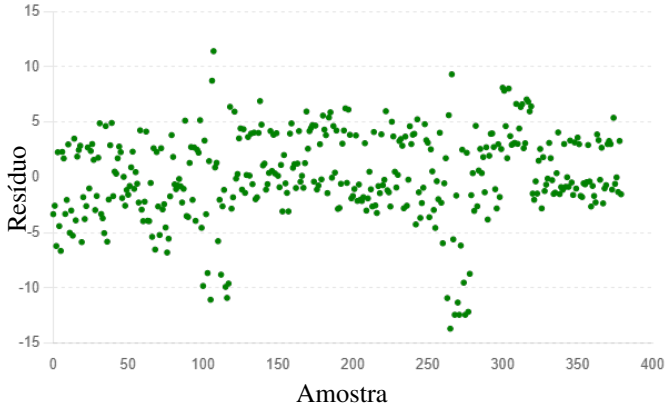

Figura 18: Resíduos de predição de k sem entrada otimizada

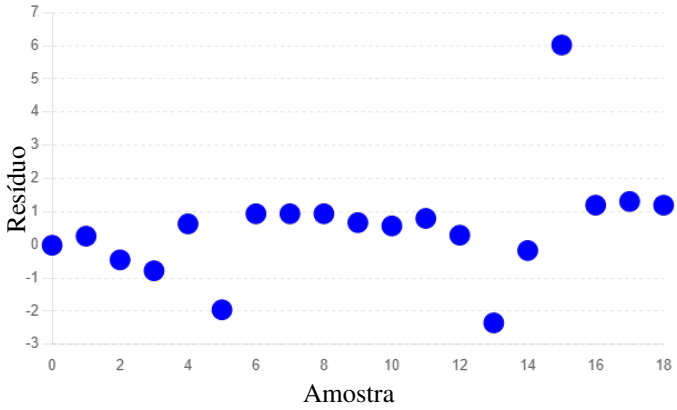

Figura 19: Resíduos de predição de a com entrada otimizada

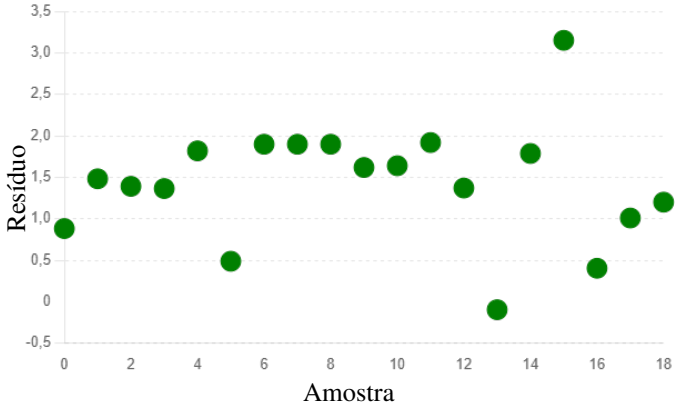

Figura 20: Resíduos de predição de k com entrada otimizada

Observando os gráficos de resíduos, nota-se uma significativa diferença na dimensão dos erros. Ao limitar os dados de entrada aos dados ótimos, o modelo tende a diminuir a intensidade de seus erros, demonstrando maior estabilidade e precisão nas predições.

Com isso, podemos concluir que a qualidade dos dados de entrada é crucial para o desempenho do modelo. Dados bem selecionados e otimizados permitem ao modelo gerar predições mais precisas e reduzir os erros de forma significativa. Isso não só melhora a confiança nas predições, mas também aumenta a robustez e a estabilidade do modelo em situações práticas.

Além disso, a análise dos resíduos confirma que o refinamento dos dados de entrada leva a uma distribuição de erros mais homogênea e menor variância. Dessa forma, modelos treinados com dados de alta qualidade são mais adequados para aplicações que exigem alta precisão e confiabilidade.

Por fim, a abordagem de utilizar dados ótimos pode ser aplicada em diversos contextos de modelagem preditiva, inclusive na modelagem de motores elétricos, reforçando a importância de uma preparação cuidadosa dos dados como parte integral do processo de desenvolvimento de modelos de *machine learning*.

V. CONCLUSÃO

Este trabalho apresentou o desenvolvimento e a aplicação de uma metodologia baseada em inteligência artificial para a identificação de parâmetros eletromagnéticos em motores elétricos. A utilização de redes neurais, especificamente do tipo Perceptron Multicamadas (MLP), mostrou-se eficaz na modelagem de relações não lineares complexas, essenciais para a previsão precisa desses parâmetros.

O processo de criação e ajuste do modelo envolveu a geração de um conjunto de dados a partir de simulações realizadas no software FEMM. As curvas de fluxo magnético por corrente obtidas para diversos materiais ferromagnéticos serviram como base para o treinamento das redes neurais. Com um treinamento rigoroso e avaliação contínua, foi possível desenvolver uma rede neural capaz de prever com precisão os coeficientes das curvas BH com um número mínimo de pontos.

Os resultados indicam que a abordagem proposta não apenas é viável, mas também pode proporcionar melhorias significativas na eficiência e no desempenho de motores elétricos. A capacidade de identificar com precisão os parâmetros eletromagnéticos permite otimizações que resultam em motores mais eficientes e com melhores características de desempenho. Além disso, nossa abordagem permite rodar o modelo neural em um microcontrolador, graças à baixa complexidade da rede neural utilizada, possibilitando a predição desses parâmetros de forma totalmente offline, mesmo com baixa capacidade computacional.

Os desafios enfrentados, como a variabilidade significativa nos dados e a necessidade de gerar dados sintéticos para complementar o treinamento, foram superados através de ajustes na metodologia e na arquitetura da rede neural.

Para trabalhos futuros, recomenda-se a expansão do conjunto de dados, incluindo uma maior variedade de materiais e modelagens de motores, permitindo uma generalização mais abrangente da rede. Além disso, a aplicação de outras técnicas de aprendizado de máquina, como as redes neurais inspiradas em física (PINN - *Physics-Informed Neural Networks*), pode trazer melhorias significativas na capacidade da rede neural de aprender o comportamento das curvas características dos motores de forma mais generalista. A implementação do modelo desenvolvido em sistemas embarcados e sua validação em aplicações reais constituem passos importantes para a consolidação e aplicação prática das contribuições deste estudo.

Conclui-se que a aplicação de inteligência artificial na identificação de parâmetros eletromagnéticos de motores elétricos é uma abordagem promissora, com potencial para revolucionar o campo e contribuir significativamente para avanços tecnológicos na indústria.

VI. AGRADECIMENTOS

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq-Brasil, pelo financiamento deste projeto (407948/2022-8) via chamada CNPq/MCTI/SEMPI Nº 56/2022.

REFERÊNCIAS

[1] B. Štumberger, et al., "Evaluation of saturation and cross-magnetization effects in interior permanent-magnet synchronous motor,"*IEEE Transactions on Industry Applications*, vol. 39, no. 5, pp. 1264-1271, 2003.

[2] G. Štumberger, et al., "Experimental method for determining magnetically nonlinear characteristics of electric machines,"*IEEE Transactions on Magnetics*, vol. 44, no. 11, pp. 4341-4344, 2008.

[3] L. Peretti, et al., "Self-commissioning of flux linkage curves of synchronous reluctance machines in quasi-standstill condition,"*IET Electric Power Applications*, vol. 9, no. 9, pp. 642-651, 2015.

[4] M. Hinkkanen, et al., "Sensorless self-commissioning of synchronous reluctance motors at standstill,"*IEEE Transactions on Industry Applications*, vol. 52, no. 4, pp. 3083-3092, 2016.

[5] B. Huang and J. Wang, "Applications of Physics-Informed Neural Networks in Power Systems - A Review,"*IEEE Transactions on Power Systems*, vol. 38, no. 1, pp. 572-584, January 2023. doi: 10.1109/TPWRS.2022.3162473.

[6] FEMM, "Soft Magnetic Materials,"[Online]. Available: `https://www.femm.info/wiki/SoftMagneticMaterials`. [Accessed: 13-Nov-2023].

[7] D. K. *Rao, SMAG Handbook: Properties of Soft Magnetic Materials*, Version 7, Frisco, TX, USA: MagWeb USA, Aug. 2021.

[8] AK Steel, *Selection of Electrical Steels for Magnetic Cores*, West Chester, OH, USA: AK Steel, 2007.

Apêndice A - Dados *Curve Fitting*

Material	a_{fit}	b_{fit}
1006 Steel	16.62	3.003
1010 Steel	19.16	3.18
1018 Steel	20.26	4.232
1020 Steel	12.19	3.97
1117 Steel	20.8	2.973
M-15 Steel	25.16	2.474
M-19 Steel	25.16	2.474
M-22 Steel	25.16	2.474
M-27 Steel	23.22	2.534
M-36 Steel	23.05	2.514
M-43 Steel	23.36	2.486
M-45 Steel	21.45	2.557
M-14	36.89	2.673
M-19	26.93	3.138
M-22	26.28	3.032
M-27	25.9	2.907

Tabela A.1: Tabela dos coeficientes para interpolação das curvas dos dos materiais

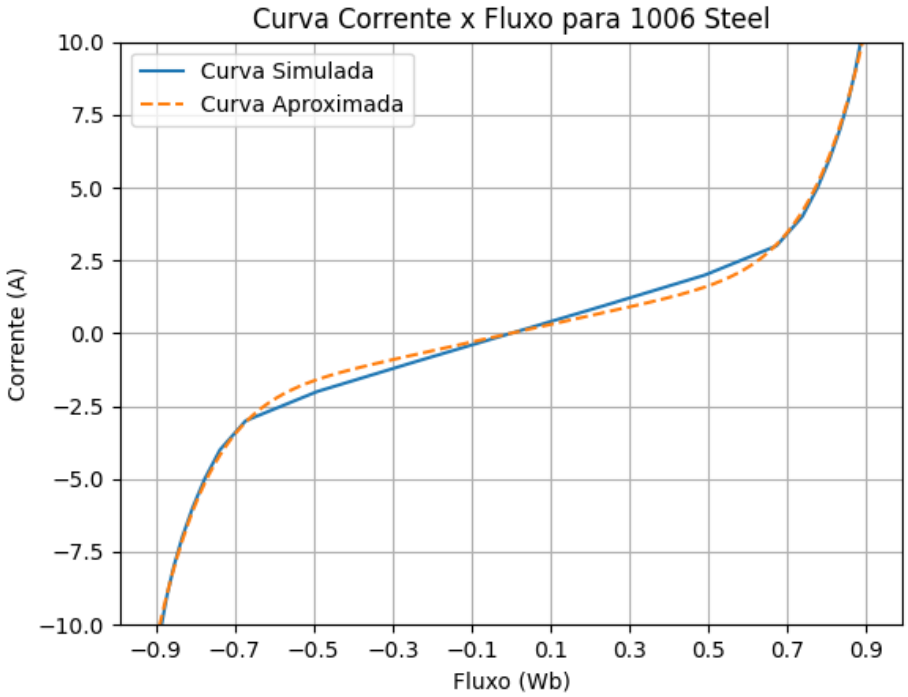

Figura A.1: Curva Simulada X Curva Aproximada 1006 Steel

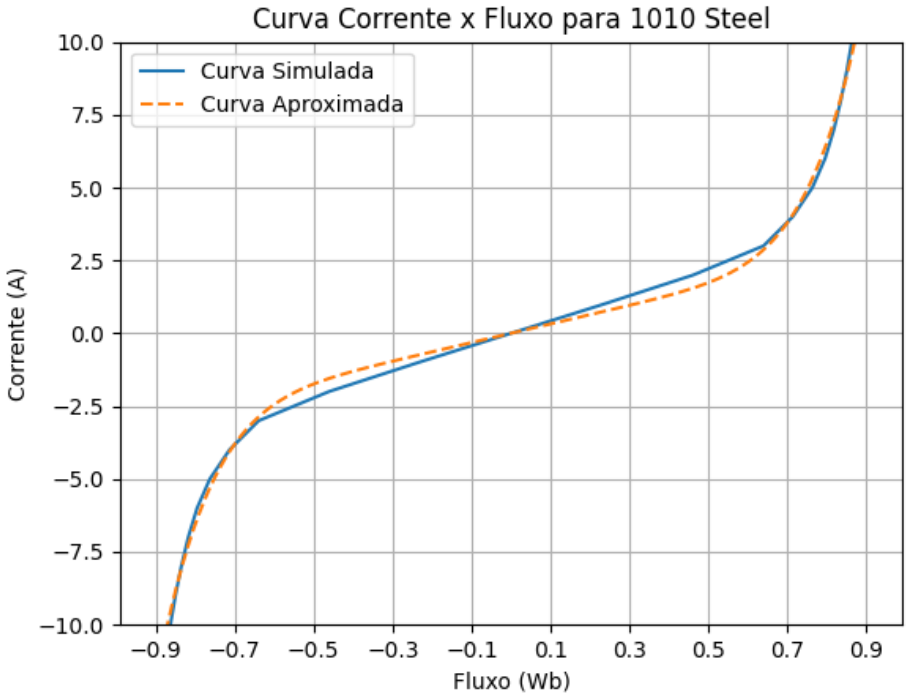

Figura A.2: Curva Simulada X Curva Aproximada 1010 Steel

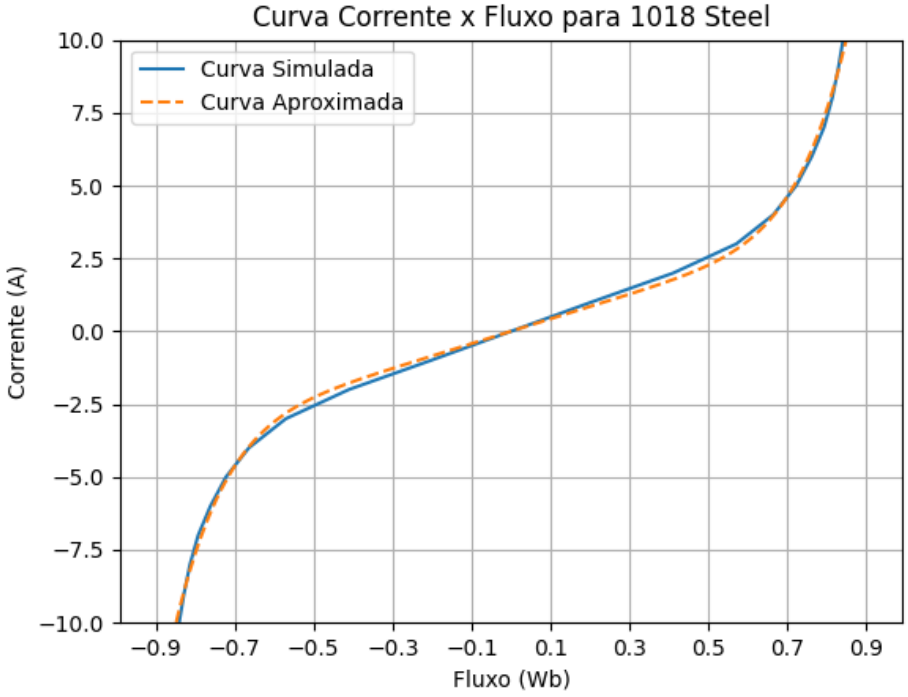

Figura A.3: Curva Simulada X Curva Aproximada 1018 Steel

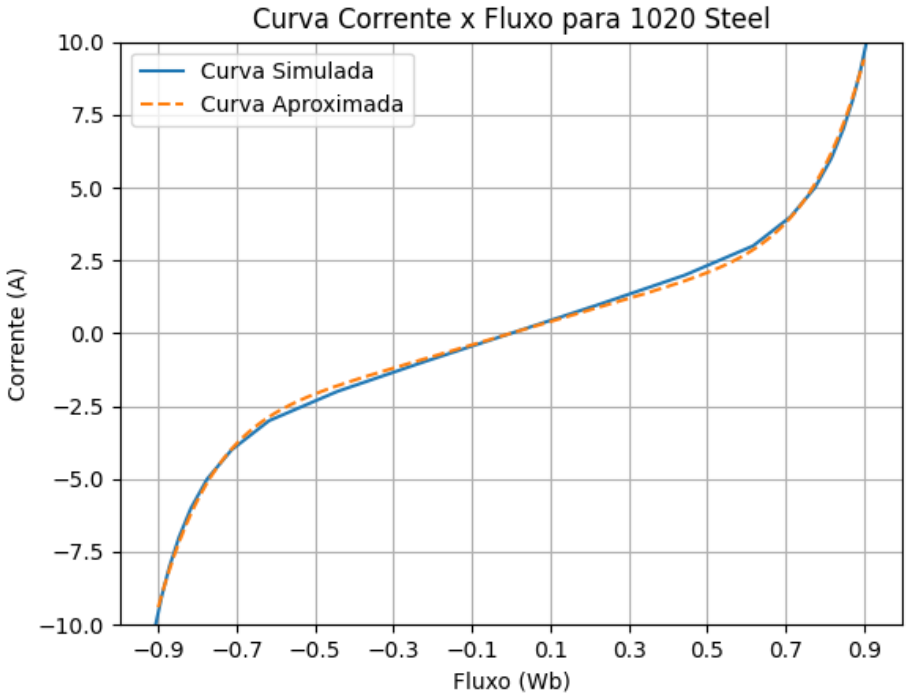

Figura A.4: Curva Simulada X Curva Aproximada 1020 Steel

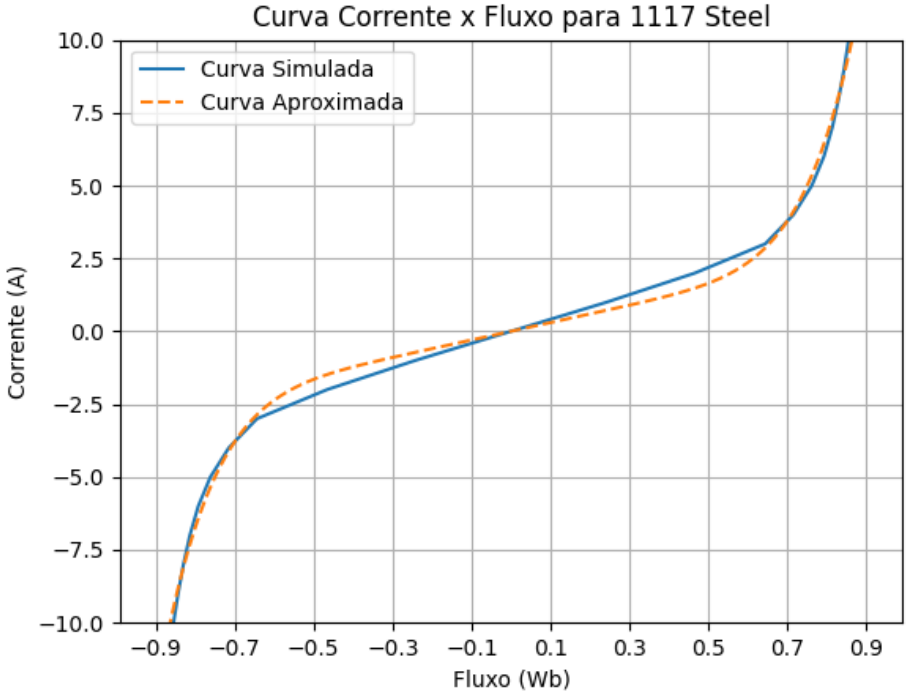

Figura A.5: Curva Simulada X Curva Aproximada 1117 Steel

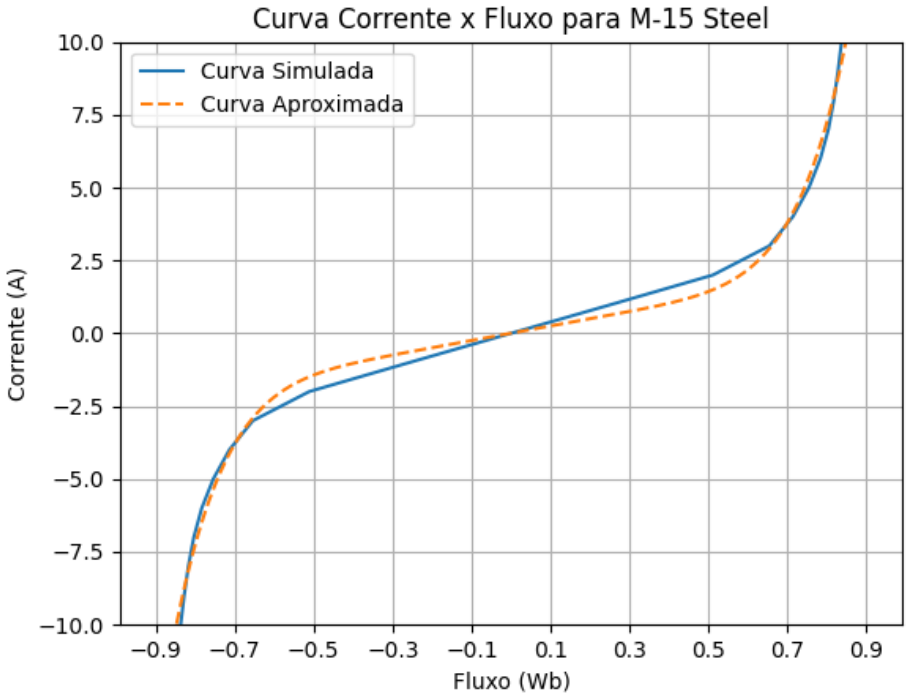

Figura A.6: Curva Simulada X Curva Aproximada M-15 Steel

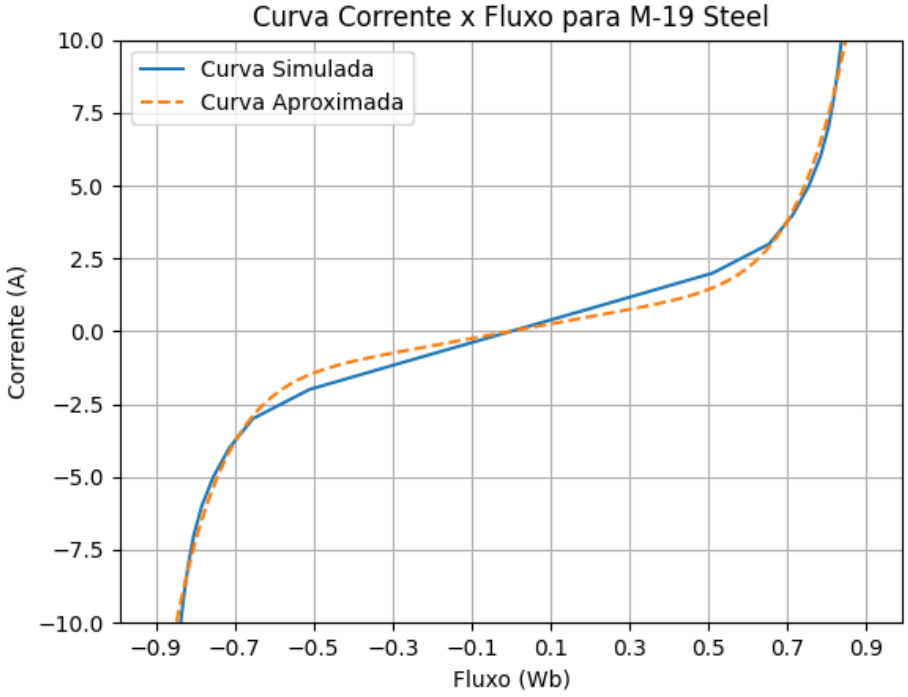

Figura A.7: Curva Simulada X Curva Aproximada M-19 Steel

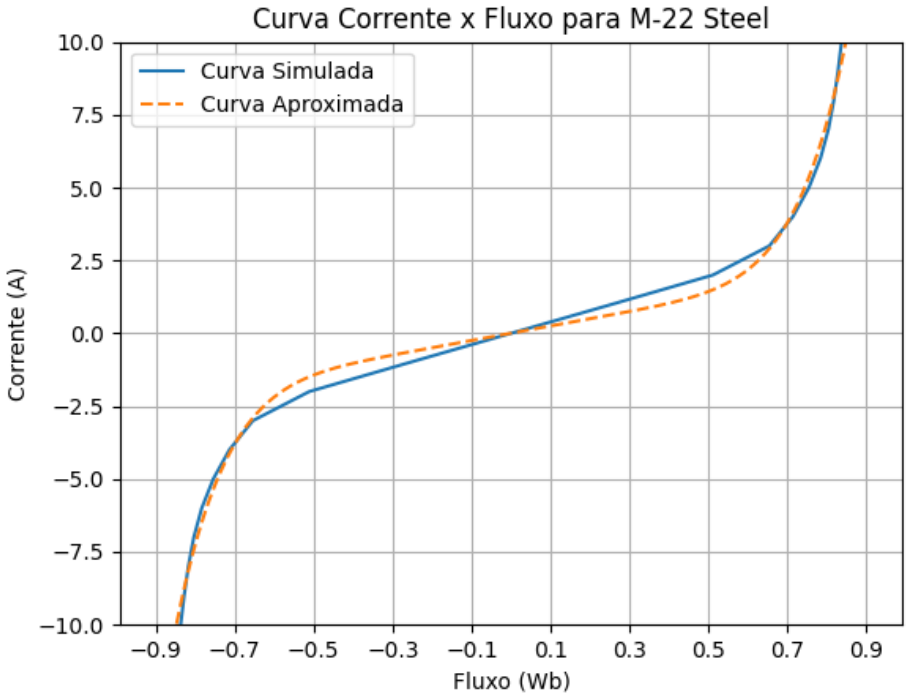

Figura A.8: Curva Simulada X Curva Aproximada M-22 Steel

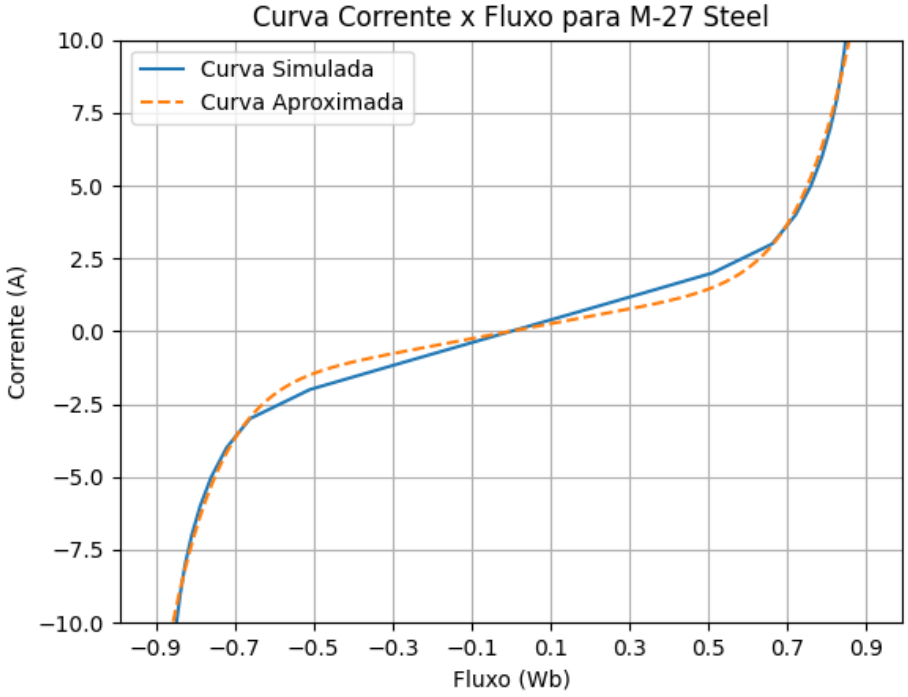

Figura A.9: Curva Simulada X Curva Aproximada M-27 Steel

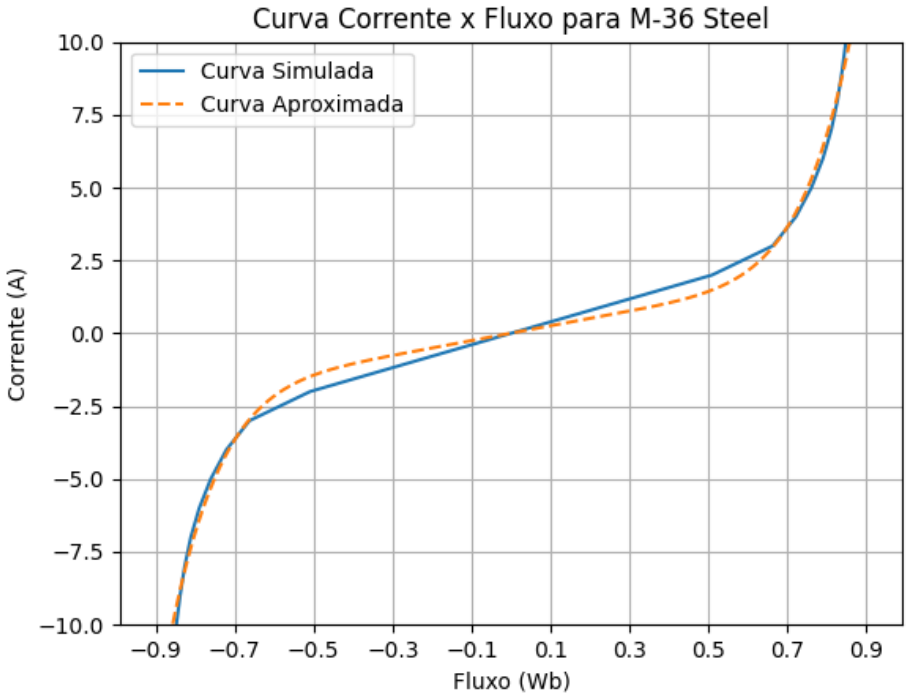

Figura A.10: Curva Simulada X Curva Aproximada M-36 Steel

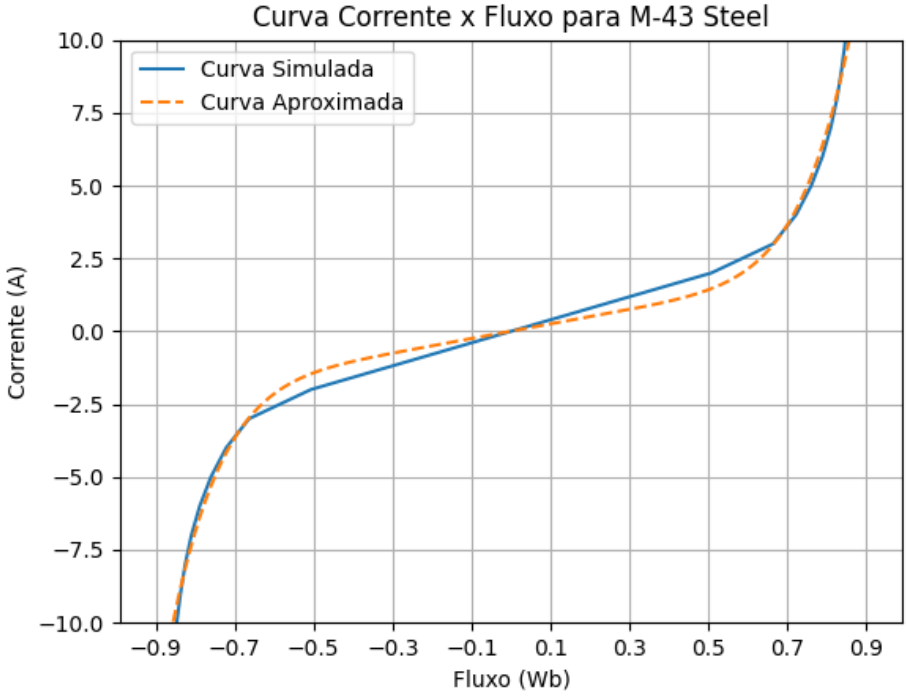

Figura A.11: Curva Simulada X Curva Aproximada M-43 Steel

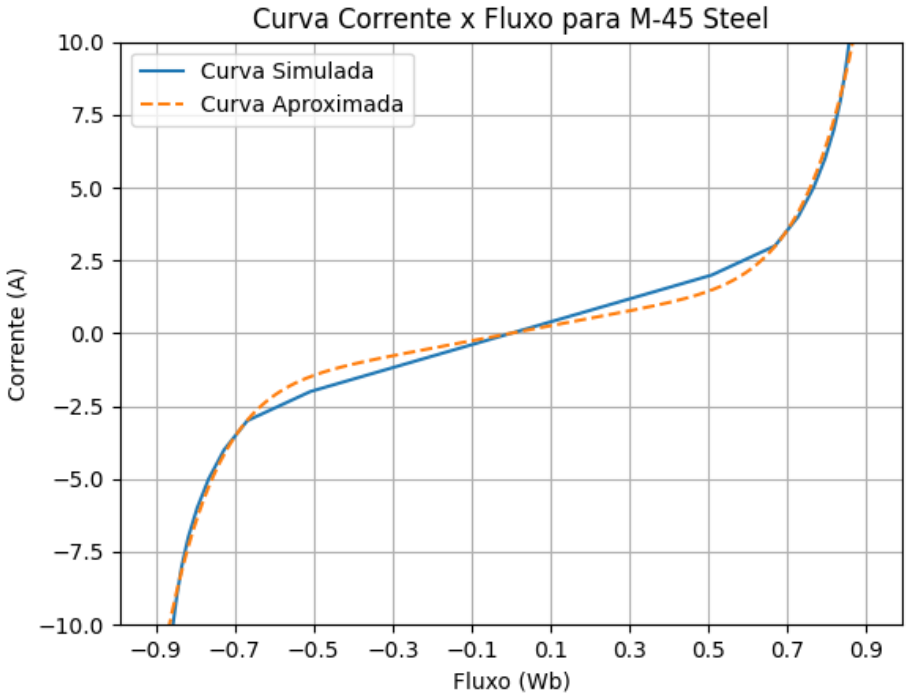

Figura A.12: Curva Simulada X Curva Aproximada M-45 Steel

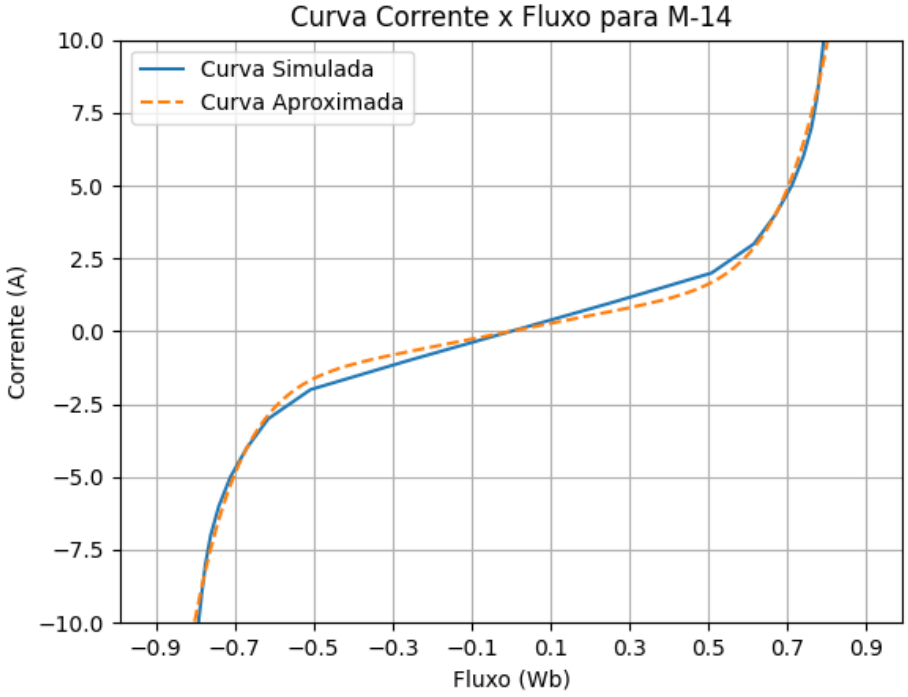

Figura A.13: Curva Simulada X Curva Aproximada M-14

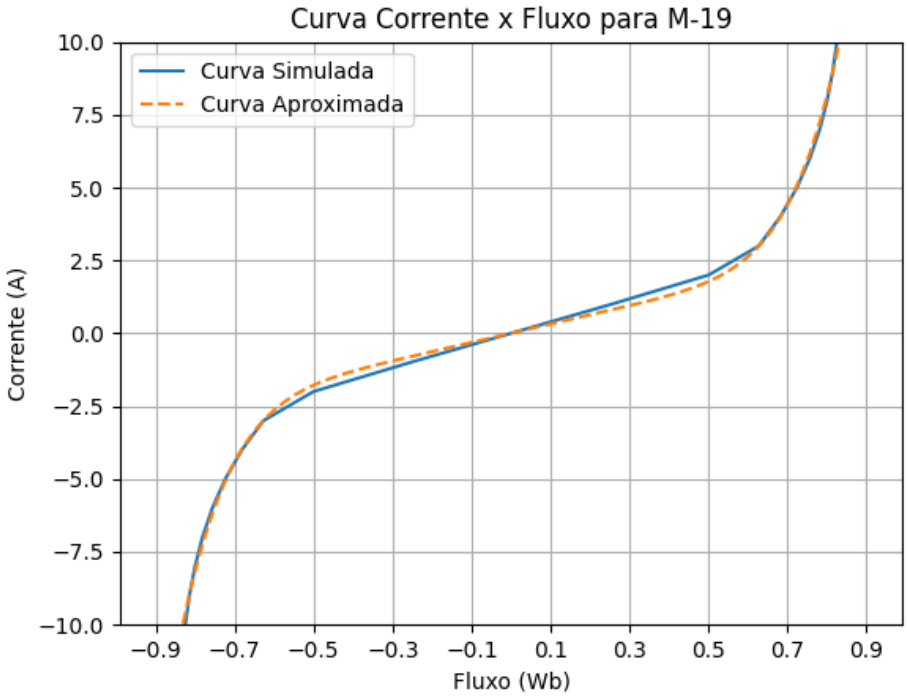

Figura A.14: Curva Simulada X Curva Aproximada M-19

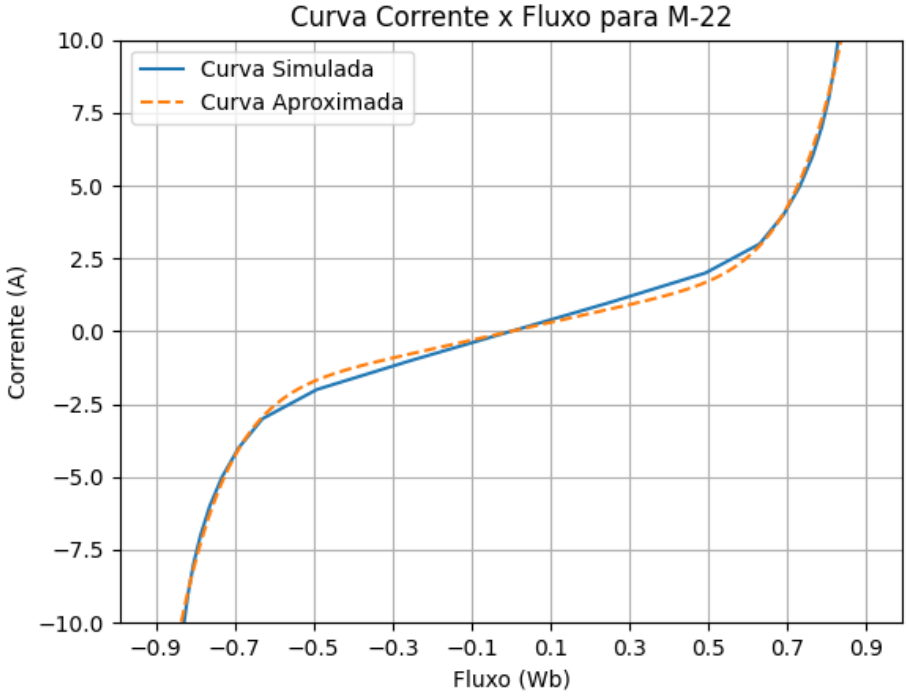

Figura A.15: Curva Simulada X Curva Aproximada M-22

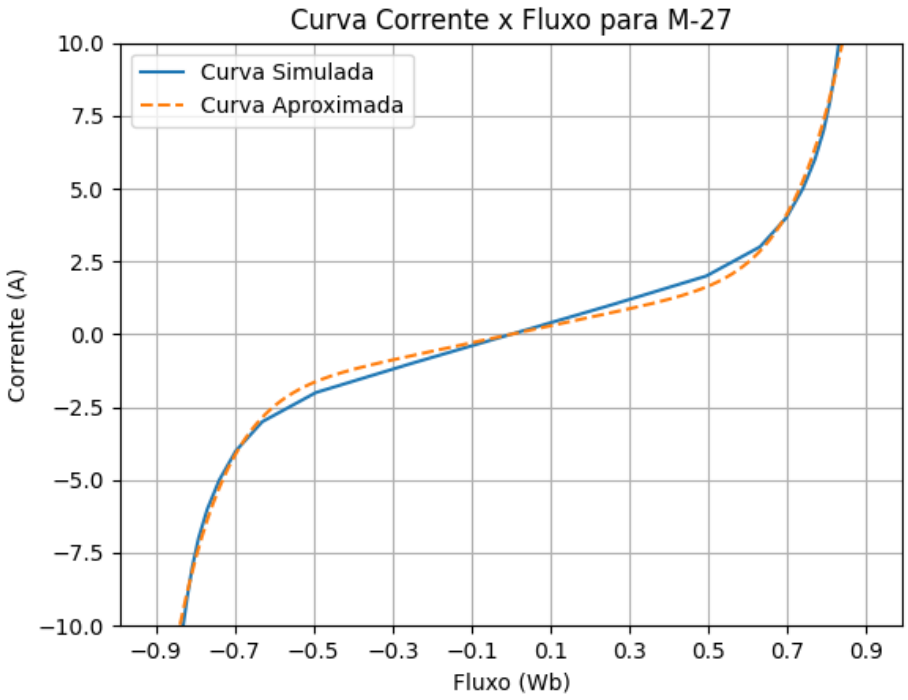

Figura A.16: Curva Simulada X Curva Aproximada M-27