



**UNIVERSIDADE FEDERAL DO CEARÁ**  
**CAMPUS SOBRAL**  
**DEPARTAMENTO DE ENGENHARIA ELÉTRICA**  
**CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA**

**IGOR SOUZA NASCIMENTO**

**COMPARAÇÃO ENTRE MÉTODOS DE TREINAMENTO DE REDES NEURAIAS  
ARTIFICIAIS NA TAREFA DE IDENTIFICAÇÃO DE SISTEMAS**

**SOBRAL**

**2023**

IGOR SOUZA NASCIMENTO

COMPARAÇÃO ENTRE MÉTODOS DE TREINAMENTO DE REDES NEURAIIS  
ARTIFICIAIS NA TAREFA DE IDENTIFICAÇÃO DE SISTEMAS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia Elétrica do Campus Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia Elétrica.

Orientador: Prof. Me. David Nascimento Coelho

SOBRAL

2023

Dados Internacionais de Catalogação na Publicação  
Universidade Federal do Ceará  
Sistema de Bibliotecas  
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

---

N195c Nascimento, Igor Souza.

Comparação entre métodos de treinamento de redes neurais artificiais na tarefa de identificação de sistemas / Igor Souza Nascimento. – 2023.  
80 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Sobral, Curso de Engenharia Elétrica, Sobral, 2023.

Orientação: Prof. Me. David Nascimento Coelho.

1. Identificação de sistemas. 2. Redes neurais artificiais. 3. Perceptron multicamadas. 4. Máquina de aprendizado extremo. 5. mínimos quadrados recursivos. I. Título.

CDD 621.3

---

IGOR SOUZA NASCIMENTO

COMPARAÇÃO ENTRE MÉTODOS DE TREINAMENTO DE REDES NEURAIAS  
ARTIFICIAIS NA TAREFA DE IDENTIFICAÇÃO DE SISTEMAS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia Elétrica do Campus Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia Elétrica.

Aprovada em: 18/07/2023

BANCA EXAMINADORA

---

Prof. Me. David Nascimento Coelho (Orientador)  
Universidade Federal do Ceará (UFC)

---

Prof. Dr. Ícaro Bezerra Viana  
Universidade Federal do Ceará (UFC)

---

Me. Pablo Eduardo Espinoza Lara  
Geoazur Laboratory (Université Côte d'Azur)

A minha mãe Sheila Maria *in memoriam* que com seu esforço e dedicação me educou e me deu a oportunidade de concluir mais uma etapa.

## AGRADECIMENTOS

A Deus por ser o autor e estar presente em nossas vidas.

A minha mãe Sheila Maria *in memoriam* que com seu esforço e dedicação me ofertou as melhores condições possíveis e me possibilitou concluir essa etapa. Sem seu amor jamais estaria nessa posição de realizar esse trabalho e concluir mais uma etapa. Um dos seus desejos era de me ver finalizar o curso de graduação. Infelizmente, em 09/04/2023 após uma longa batalha contra um câncer, ela não está presente fisicamente, mas sempre estará comigo me ajudando em tudo que eu fizer. Esse trabalho é uma forma singela de agradecimento e reconhecimento pelo seu esforço.

Ao orientador David que me acolheu e me guiou para a realização desse trabalho. Pela sua paciência e companheirismo durante um período difícil da minha vida em que passei nos últimos anos. As reuniões quase que semanais para discussões do tema eram também conversas que me ajudavam a não baixar a auto estima e seguir em busca da conclusão do curso.

A minha irmã Ingrid, meu pai Roberto, minha namorada Juliana e minha avó Maria Peixoto que sempre foram presentes e importantes na caminhada.

Ao amigo Carlos André que também me ajudou no momento mais difícil no qual queria desistir e me aconselhou a seguir em frente.

Não posso deixar de lembrar da minha prima Loyse *in memoriam* que fez parte do meu crescimento e amadurecimento e que também partiu de forma melancólica no ano de 2022.

"A morte deixa uma mágoa que ninguém pode curar, o amor deixa uma memória que ninguém pode roubar."

(Khalil Gibran)

## RESUMO

Neste trabalho, diferentes algoritmos de treinamento das redes neurais artificiais do tipo *perceptron* multicamadas são comparadas na tarefa de identificação de sistemas. Para gerar o banco de dados, foi realizada a simulação, no software Matlab, de um motor de corrente contínua sujeito a diferentes níveis de tensão e carga, e, como saídas, foram consideradas a corrente e a velocidade deste motor. Devido à simplicidade, a arquitetura da rede neural escolhida possui apenas uma camada oculta, e a comparação foi realizada entre os métodos *backpropagation*, máquina de aprendizado extremo em batelada e máquina de aprendizado extremo com treinamento online. Todos os modelos foram implementados no *software* Matlab e, a partir das medidas de erro quadrático médio entre os valores reais e preditos, o treinamento online com máquina de aprendizado extremo se mostrou o mais eficaz na tarefa de identificação.

**Palavras-chave:** Identificação de sistemas; retropropagação do erro; Redes Neurais Artificiais; Perceptron Multicamadas; Máquina de Aprendizado Extremo; mínimos quadrados ordinários; mínimos quadrados recursivos.



## ABSTRACT

In this work, different algorithms for training multilayer perceptron artificial neural networks are compared in the system identification task. To generate the database, a simulation, in Matlab software, of a direct current motor subject to different levels of voltage and load was carried out, and the current and speed of this motor are considered as outputs. Due to its simplicity, the chosen neural network architecture has only one hidden layer, and the comparison was performed between the methods *error backpropagation*, extreme learning machine in batch and extreme learning machine with online training. All models were implemented in Matlab *software* and, based on the measurements of mean squared error between real and predicted values, the online training with an extreme learning machine was the most effective in the identification task.

**Keywords:** System Identification; backpropagation; Artificial Neural Networks; Multilayer Perceptron; Extreme Learning Machine; ordinary least squares; recursive least squares.

## LISTA DE FIGURAS

Figura 1 – Sinal Aleatório . . . . .	26
Figura 2 – Sinal PRBS . . . . .	27
Figura 3 – Sinal APRBS . . . . .	28
Figura 4 – Função Sigmóide . . . . .	29
Figura 5 – Função Tangente Hiperbólica . . . . .	29
Figura 6 – MLP de 2 camadas ocultas . . . . .	30
Figura 7 – Diagrama de blocos do motor CC no Simulink com entradas APRBS . . . . .	38
Figura 8 – Diagrama de blocos do motor CC no Simulink com entradas aleatórias . . . . .	39
Figura 9 – Predição velocidade MLP teste 01 . . . . .	44
Figura 10 – Predição corrente MLP teste 01 . . . . .	44
Figura 11 – Predição velocidade teste 02 . . . . .	45
Figura 12 – Predição corrente teste 02 . . . . .	46
Figura 13 – Predição velocidade teste 03 . . . . .	47
Figura 14 – Predição corrente teste 03 . . . . .	48
Figura 15 – Predição velocidade teste 04 . . . . .	49
Figura 16 – Predição corrente teste 04 . . . . .	49
Figura 17 – Predição velocidade teste 05 . . . . .	51
Figura 18 – Predição corrente teste 05 . . . . .	51
Figura 19 – Predição velocidade teste 06 . . . . .	52
Figura 20 – Predição corrente teste 06 . . . . .	53
Figura 21 – Predição velocidade teste 07 . . . . .	54
Figura 22 – Predição corrente teste 07 . . . . .	54
Figura 23 – Predição velocidade teste 08 . . . . .	55
Figura 24 – Predição corrente teste 08 . . . . .	56
Figura 25 – Predição velocidde teste 09 . . . . .	57
Figura 26 – Predição corrente teste 09 . . . . .	57
Figura 27 – Predição velocidade teste 10 . . . . .	60
Figura 28 – Predição corrente teste 10 . . . . .	60
Figura 29 – Predição velocidade teste 11 . . . . .	61
Figura 30 – Predição corrente teste 11 . . . . .	62
Figura 31 – Predição velocidade teste 12 . . . . .	63

Figura 32 – Predição corrente teste 12 . . . . .	64
Figura 33 – Predição velocidade teste 13 . . . . .	65
Figura 34 – Predição corrente teste 13 . . . . .	65
Figura 35 – Predição velocidade teste 14 . . . . .	67
Figura 36 – Predição corrente teste 14 . . . . .	67
Figura 37 – Predição velocidade teste 15 . . . . .	68
Figura 38 – Predição corrente teste 15 . . . . .	69
Figura 39 – Predição velocidade teste 16 . . . . .	70
Figura 40 – Predição corrente teste 16 . . . . .	70
Figura 41 – Predição velocidade teste 17 . . . . .	71
Figura 42 – Predição corrente teste 17 . . . . .	72
Figura 43 – Predição velocidade teste 18 . . . . .	73
Figura 44 – Predição corrente teste 18 . . . . .	73

## LISTA DE TABELAS

Tabela 1 – Parâmetros do Motor CC . . . . .	40
Tabela 2 – Parâmetros padrão para as simulações . . . . .	41
Tabela 3 – Hiper parâmetros MLP . . . . .	42
Tabela 4 – Hiper Parâmetros ELM Não Recursiva . . . . .	42
Tabela 5 – Hiper parâmetros ELM Recursiva . . . . .	42
Tabela 6 – Resumo dos testes MLP com entradas aleatórias . . . . .	46
Tabela 7 – Resumo dos testes ELM Não recursivo entradas aleatórias . . . . .	50
Tabela 8 – Resumo dos teste ELM Recursivo com entradas aleatórias . . . . .	58
Tabela 9 – Resumo dos teste com entradas aleatórias . . . . .	59
Tabela 10 – Resumo dos teste MLP com entradas APRBS . . . . .	62
Tabela 11 – Resumo dos teste ELM Não recursivo com entradas APRBS . . . . .	66
Tabela 12 – Resumo dos teste ELM Recursivo com entradas APRBS . . . . .	74
Tabela 13 – Resumo dos testes com entradas APRBS . . . . .	75
Tabela 14 – Resumo dos testes com entradas Aleatórias e APRBS . . . . .	75

## LISTA DE ABREVIATURAS E SIGLAS

<i>ELM<sub>b</sub></i>	<i>Extreme Learning Machine Batch</i>
<i>ELM<sub>o</sub></i>	<i>Extreme Learning Machine Online</i>
APRBS	<i>Amplitude Modulated Pseudo Random Binary Sequence</i>
AR	<i>Autoregressive</i>
ARX	<i>Autoregressive with eXogenous Variables</i>
BP	<i>Error Backpropagation</i>
ELM	<i>Extreme Learning Machine</i>
MLP	<i>Multilayer Perceptron</i>
MSE	Mean Squared Error
NARX	<i>Nonlinear AutoRegressive with eXogenous inputs</i>
OLS	<i>Ordinary Least Squares</i>
PRBS	<i>Pseudo Random Binary Sequence</i>
RLS	<i>Recursive Least Squares</i>
RNA	Redes Neurais Artificiais

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>15</b>
<b>1.1</b>	<b>Objetivos</b>	<b>16</b>
<i>1.1.1</i>	<i>Objetivos Gerais</i>	<i>16</i>
<i>1.1.2</i>	<i>Objetivos Específicos</i>	<i>16</i>
<b>1.2</b>	<b>Justificativa</b>	<b>16</b>
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b>	<b>18</b>
<b>2.1</b>	<b>IDENTIFICAÇÃO DE SISTEMAS</b>	<b>18</b>
<i>2.1.1</i>	<i>Aquisição de Dados</i>	<i>19</i>
<i>2.1.2</i>	<i>Definição da Estrutura do Modelo</i>	<i>19</i>
<i>2.1.3</i>	<i>Estimação Linear dos Parâmetros de um Sistema</i>	<i>21</i>
<i>2.1.4</i>	<i>Mínimos Quadrados Batelada</i>	<i>21</i>
<i>2.1.5</i>	<i>Mínimos Quadrados Recursivos</i>	<i>23</i>
<b>2.2</b>	<b>SINAIS DE ENTRADA</b>	<b>25</b>
<i>2.2.1</i>	<i>Sinais Aleatórios</i>	<i>26</i>
<i>2.2.2</i>	<i>Sequência Binária Pseudoaleatória</i>	<i>26</i>
<i>2.2.3</i>	<i>Sequência Binária Pseudoaleatória Modulada em Amplitude</i>	<i>27</i>
<b>2.3</b>	<b>REDES NEURAIIS ARTIFICIAIS</b>	<b>28</b>
<i>2.3.1</i>	<i>Conceitos Básicos</i>	<i>28</i>
<i>2.3.1.1</i>	<i>Função de ativação</i>	<i>28</i>
<i>2.3.2</i>	<i>Perceptron Multicamadas</i>	<i>29</i>
<i>2.3.2.1</i>	<i>Princípios de Funcionamento</i>	<i>30</i>
<i>2.3.3</i>	<i>Treinamento de uma MLP</i>	<i>31</i>
<i>2.3.4</i>	<i>Máquina de Aprendizado Extremo</i>	<i>33</i>
<i>2.3.4.1</i>	<i>ELM Não Recursivo</i>	<i>33</i>
<i>2.3.4.2</i>	<i>ELM Recursivo</i>	<i>35</i>
<b>2.4</b>	<b>Validação do Modelo</b>	<b>36</b>
<i>2.4.1</i>	<i>Medida de Erro - Erro Quadrático Médio</i>	<i>37</i>
<b>3</b>	<b>MATERIAIS E MÉTODOS</b>	<b>38</b>
<b>3.1</b>	<b>Sistema Utilizado</b>	<b>38</b>
<b>3.2</b>	<b>Banco de dados</b>	<b>40</b>

3.3	<b>Organização dos experimentos</b>	40
3.4	<b>Redes neurais – Seleção do Modelo</b>	41
3.4.1	<i>MLP</i>	41
3.4.2	<i>ELM Batelada</i>	42
3.4.3	<i>ELM Online</i>	42
4	<b>RESULTADOS E DISCUSSÕES</b>	43
4.1	<b>Sinal de entrada de tensão e carga Aleatórios</b>	43
4.1.1	<i>MLP</i>	43
4.1.2	<i>ELM Não recursivo</i>	47
4.1.3	<i>ELM Online</i>	50
4.2	<b>Sinal de entrada de tensão e carga APRBS</b>	59
4.2.1	<i>ELM Não recursivo</i>	63
4.2.2	<i>ELM Online</i>	66
4.2.3	<i>Resumo das simulações</i>	75
5	<b>CONCLUSÕES E TRABALHOS FUTUROS</b>	77
	<b>REFERÊNCIAS</b>	78
	<b>APÊNDICES</b>	81
	<b>APÊNDICE A – Algoritmo de geração do sinal APRBS</b>	81
	<b>ANEXOS</b>	81

## 1 INTRODUÇÃO

Entender os sistemas físicos, químicos e biológicos que representam os diversos fenômenos da natureza é fundamental para a sociedade. Os sistemas reais podem ser representados matematicamente, se conhecidos as leis que regem sua dinâmica, ou podem ser modelados através de técnicas de identificação de sistemas. A identificação de sistemas ou modelagem empírica são métodos que coletam os dados diretamente dos sistemas, e, a partir de tais observações, desenvolvem modelos matemáticos capazes de explicar estes dados (AGUIRRE, 2015). Embora seja de fundamental importância para o projeto, controle e análise de processos industriais em geral, esta é uma tarefa desafiadora. Sistemas modernos podem apresentar uma ampla classe de não linearidades (BILLINGS, 2013), e, além disso, geram uma quantidade de dados cada vez maior, o que demanda computadores e algoritmos mais robustos para processamento.

Devido a estas complexidades, diversos algoritmos de aprendizado de máquina já foram aplicados na solução desta tarefa. Regressão baseada em vetores suporte (GRETTON *et al.*, 2001), (MARTINEZ-RAMON *et al.*, 2006), métodos de kernel (CHIUSO; PILLONETTO, 2019) e processos gaussianos (GREGORÄIÄ; LIGHTBODY, 2008) são alguns exemplos que podem ser citados. Além destes, as redes neurais artificiais vem sendo muito utilizadas para abordar o problema de identificação de sistema (PATTANAIAK *et al.*, 2023), visto que estas são conhecidas como aproximadoras universais de funções (HORNIK *et al.*, 1989). Manipuladores industriais (DINARY, 2015), biorreatores e colunas de destilação (PRASAD; BEQUETTE, 2003) são alguns dos sistemas já abordados por esses algoritmos.

As redes neurais artificiais possuem diversas arquiteturas. As redes neurais profundas (FORGIONE; PIGA, 2021), e as redes recorrentes (BESSA; BARRETO, 2019) são algumas das arquiteturas já aplicadas em identificação de sistemas. De grande interesse deste trabalho, as redes neurais de única camada oculta possuem diversas aplicações nesta tarefa. Ye *et al.* (2013) utilizou a máquina de aprendizado extremo para previsão de consumo de eletricidade. Wei *et al.* (2019) aplicaram identificação em ambientes não-estacionários e com redes neurais variantes no tempo e adaptação do algoritmo de treinamento ELM batelada com grande quantidade de dados. Em Ye *et al.* (2013) é aplicado identificação de sistemas em ambientes industriais onde os dados são produzidos em tempo real e tem a ELM Online como treinamento mais adequado.

Mesmo as arquiteturas mais simples, além dos parâmetros que são ajustados durante o treinamento do modelo, possuem os hiperparâmetros, os quais devem ser configurados previamente e impactam diretamente na eficiência e no aprendizado do modelo (DUTTA, 2022)



## 1.1 Objetivos

### 1.1.1 *Objetivos Gerais*

O objetivo geral deste trabalho é a comparação de algoritmos de treinamento de redes neurais artificiais na tarefa de identificação de sistemas, avaliando a influência dos hiperparâmetros no erro quadrático médio entre as saídas reais e estimadas.

### 1.1.2 *Objetivos Específicos*

- Gerar bancos de dados, através de um sistema simulado, contendo diferentes sinais de entrada e saída;
- Implementar diferentes algoritmos para treinamento de redes neurais artificiais;
- Avaliar, utilizando o erro quadrático médio entre os sinais reais e os sinais estimados, diferentes combinações de hiperparâmetros dos algoritmos.

## 1.2 Justificativa

O problema de identificação de sistemas tem sido abordado nas últimas décadas por muitos pesquisadores de Redes Neurais Artificiais (RNA).

Alguns autores tem aplicado *Multilayer Perceptron* (MLP) na solução de problemas de identificação usando o algoritmo de *Error Backpropagation* (BP). Porém, o processamento de identificação de sistemas usando MLP e o treinamento BP é lento (PATTANAİK *et al.*, 2023).

Ao contrário das RNA com aprendizado BP, o *Extreme Learning Machine* (ELM) é significativamente mais eficiente e tem maior tendência para alcançar resultados melhores com uma vasta gama de aplicações (ALBADR; TIUNA, 2017). Algoritmos de Aprendizagem Extrema possibilitam a flexibilidade de treinar a partir de um banco de dados em batelada ou de treinar recursivamente (LI *et al.*, 2018).

Comparar os algoritmos em vários cenários (MASHOR, 2004) de cada algoritmo de aprendizagem de redes neurais é importante para saber a robustez e *performance* de cada algoritmo, além de gerar conhecimento para aplicação de um processo de treinamento para determinado sistema de acordo com as características para obter o melhor resultado possível (KUMAR; SRIVASTAVA, 2019).

Neste contexto, Kumar e Srivastava (2019) comparou 3 arquiteturas de redes neurais

multicamadas, 1 arquitetura de rede recorrente e 1 rede neural NARX. Já Tiwari *et al.* (2013) compararam métodos de treinamento *backpropagation* para identificação de sistemas de potência. Por fim, Mashor (2004) comparou uma MLP híbrida com a MLP e 1 rede recorrente para identificação *online* de sistemas.

Assim, neste trabalho, 3 métodos de treinamento de redes neurais artificiais de camada única na tarefa de identificação de sistemas são comparados. São estes *backpropagation* e máquina de aprendizagem extrema em duas versões, batelada e online. Estes foram escolhidos pelas suas arquiteturas simples e que podem ser, por exemplo, aplicados para simular ou calcular controladores para diversos tipos de sistemas.

## 2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, os principais aspectos teóricos para o entendimento deste trabalho são discutidos. Os tópicos de identificação de sistemas, geração de sinais e RNA são os assuntos discutidos.

### 2.1 IDENTIFICAÇÃO DE SISTEMAS

A identificação de sistemas é um problema de otimização que envolve algumas medidas para adequação de modelos a representar um processo real. Fundamentalmente, identificação de sistemas consiste na determinação de um modelo matemático que represente aspectos essenciais do sistema, caracterizado pela manipulação dos sinais de entrada e saída e que estão relacionados através de uma função transferência contínua ou discreta, (AGUIRRE, 2015). A escolha de modelos e seus respectivos ajustes são influenciados por diversos fatores, dentre os quais podem ser citados :

- Linearidade ou não linearidade;
- Complexidade do modelo;
- Seleção da medida de erro a ser minimizada;
- Presença de ruídos.

As técnicas de modelagem podem ser classificadas em 3 grandes grupos chamados de modelagem caixa-branca, modelagem caixa preta e modelagem caixa-cinza (AGUIRRE, 2015):

- Modelagem Caixa Preta: esta modelagem tem como característica pouco ou nenhum conhecimento do sistema em análise. Nesse tipo de modelo apenas entrada e saída do sistema são necessárias durante a identificação.
- Modelagem Caixa-branca: esta modelagem exige conhecimento profundo do sistema, assim como a leis da física e da química que regem o sistema a ser modelado. Dessa forma a função matemática que descreve o comportamento é previamente conhecida.
- Modelagem Caixa-cinza: segundo Aguirre (2015), esta modelagem busca mesclar as vantagens dos procedimentos caixa preta e caixa-branca. Portanto, essa técnica é baseada na aquisição dos dados de entrada e saída do sistema e o conhecimento complementar das leis que regem o sistema.

A identificação de um sistema, utilizando o modelo caixa-preta como referência

pode ser dividido em quatro etapas:

1. Aquisição de dados de entrada/saída através de um protocolo de experimentação;
2. Definição da estrutura do modelo. Tendo como parâmetro fundamental a ser analisado o quanto complexo ele irá ser mediante os parâmetros propostos;
3. Estimação dos parâmetros do modelo;
4. Validação do modelo identificado.

### 2.1.1 Aquisição de Dados

Como visto nos tipos de técnicas de identificação, todas elas propõem a obtenção dos modelos a partir de dados. Esses dados podem ser gerados com as excitações condizentes com o tipo do sistema e com a consequente resposta que esses sinais irão gerar.

### 2.1.2 Definição da Estrutura do Modelo

#### *Modelo Autoregressivo*

Os modelos Autorregressivo (*Autoregressive, AR*) são um tipo de modelo de série temporal usado em várias áreas da engenharia, estatística e economia. Ele utiliza valores passados obtidos durante o processo para a correção de dados. O modelo *Autoregressive (AR)* pode ser descrito genericamente a partir da equação a abaixo:

$$Y(t) = \sum_{i=1}^p a_i Y(t-i) + e(t) \quad (2.1)$$

Onde  $Y(t)$  é a variável de saída no tempo  $t$ ,  $a_i$  são os coeficientes de autorregressão,  $p$  é a ordem do modelo e o  $e(t)$  é o erro. Os coeficientes de autorregressão  $a_i$  representam a relação entre os valores da variável de saída em diferentes tempos  $t$ . A ordem do modelo é um número que representa a quantidade de valores passados que serão incluídos na previsão.

O objetivo de um modelo AR é determinar os coeficientes de autorregressão para possibilitar que o modelo seja utilizado para fazer previsões com base em valores passados.

#### *Modelo Autoregressivo com Entradas Exógenas*

Do inglês ARX (*autoregressive with exogenous variables*) o que diferencia o modelo *Autoregressive with exogenous Variables (ARX)* do AR é a adição de variáveis externas que tem

alguma relação com a variável de saída. A parte autorregressiva de um modelo ARX é referente ao uso dos próprios valores passados da variável para prever valores futuros. Assim, o modelo ARX pode ser descrito genericamente a partir da equação a abaixo:

$$Y(t) = \sum_{i=1}^p a_i Y(t-i) + \sum_{j=1}^q b_j X(t-j) + e(t) \quad (2.2)$$

Onde  $X_1(t), \dots, X_n(t)$  são as variáveis de entrada e  $a_1, a_2, \dots, a_n$  são os coeficientes de autorregressão. Os termos  $b_1, b_2, \dots, b_q$  são os coeficientes das entradas externas.  $Y(t)$  e  $e(t)$  já foram definidos.

O objetivo de um modelo ARX é definir tanto os coeficientes autorregressivos e os coeficientes da entrada. A partir desses coeficientes, modelo é estabelecido para fazer previsões a partir de valores anteriores de saída e entradas.

#### *Modelo Autoregressivo com Média Móvel e Entradas Exógenas*

Do inglês ARMAX (*Autoregressive Moving Average with eXogenous variables*). O que diferencia do modelo ARX é a adição de uma média móvel dos erros entre os valores obtidos e os valores previstos.

$$Y(t) = \sum_{i=1}^p a_i Y(t-i) + \sum_{j=1}^q b_j X(t-j) + \sum_{k=1}^m c_k e(t-k) \quad (2.3)$$

Onde  $Y(t)$  é a variável de saída no tempo  $t$ .  $X_1(t), \dots, X_n(t)$  são as variáveis de entrada e  $a_1, a_2, \dots, a_n$  são os coeficientes de autorregressão. Os termos  $b_1, b_2, \dots, b_q$  são os coeficientes das entradas externas  $e(t)$  é o erro e os coeficientes  $c_1, c_2, \dots, c_m$  são os coeficientes da média móvel.

#### *Modelo Autoregressivo não-linear com Entradas Exógenas*

Do inglês *Nonlinear AutoRegressive with eXogenous inputs*. Diferente dos modelos descritos anteriormente, o modelo *Nonlinear AutoRegressive with eXogenous inputs* (NARX) é usado para tratar relações não-lineares. Assim como todos os outros, o modelo NARX é um modelo autorregressivo que possuem entradas que podem ou não ter relações com a saída desejada. Genericamente, a equação que descreve um modelo NARX possui duas componentes, a componente autorregressiva e componente de entrada exógena.

A componente autorregressiva captura a relação entre os valores passados e o valores atuais. A equação para a componente autorregressiva é descrita a seguir:

$$Y(t) = f(Y(t-1), Y(t-2), \dots, Y(t-n)) \quad (2.4)$$

Onde  $Y(t)$  é o valor atual da saída e  $Y(t-1), Y(t-2), \dots, Y(t-n)$  são os valores anteriores da saída, e  $f$  é uma função não linear que relaciona os valores passados com os valores atualizados. Já a componente das entradas exógenas captura a influência das variáveis externas na variável de saída. A equação para a componente das entradas externas é descrita a seguir:

$$Y(t) = f(X_1(t), X_2(t), \dots, X_n(t)) \quad (2.5)$$

Onde  $Y(t)$  é o valor atual da saída e  $X_1(t), X_2(t), \dots, X_n(t)$  são os valores atualizados das entradas e  $f$  é uma função não linear que relaciona as variáveis externas com a variável atualizada da saída.

Na próxima seção, dois modelos clássicos para estimação de parâmetros serão discutidos.

### **2.1.3 Estimação Linear dos Parâmetros de um Sistema**

Karl Friedrich Gauss formulou o Princípio Dos Mínimos Quadrados ao final do século 18 para prever a trajetória dos planetas e cometas a partir das observações realizadas (COELHO; COELHO, 2004). Basicamente, o método de mínimos quadrados determina que o valor mais provável de quantidades desconhecidas é aquela em que a soma dos quadrados das diferenças entre os valores observados e os valores calculados é mínima.

Existem dois métodos de mínimos quadrados: os mínimos quadrados recursivos e os mínimos quadrados batelada (não recursivo) que serão definidos nas próximas seções.

### **2.1.4 Mínimos Quadrados Batelada**

Também conhecido como Mínimos Quadrados Ordinários (*Ordinary Least Squares*, OLS), este algoritmo necessita que todos os dados de entrada e saída do sistema, conhecido ou não, estejam disponíveis para treinamento. Seguindo os passos conforme Aguirre (2015), supõe-se o vetor  $y$  de medidas de  $k$  elementos e  $\theta$  seja um vetor constante de elementos desconhecidos. De acordo com a definição, o objetivo é encontrar um vetor de estimativa  $\hat{\theta}$  de  $\theta$ . Assumindo

que cada elemento do vetor  $y$  é uma combinação linear de  $\theta$ , com uma adição de ruído, temos:

$$y_k = x_k^T \theta + v_k \quad (2.6)$$

Definidos os vetores  $X = (x_1 x_2 \dots x_n)$  e  $\theta = (\theta_1 \theta_2 \dots \theta_n)^T$  a equação 2.6 pode ser descrita matricialmente da seguinte forma:

$$Y = X^T \theta + \gamma \quad (2.7)$$

Definindo  $E_y$  como a diferença entre o vetor de medidas e o vetor  $X\hat{\theta}$ :

$$E_y = Y - X\hat{\theta} \quad (2.8)$$

De acordo com a definição do método de mínimos quadrados, o valor mais provável do vetor de  $\theta$  é o vetor  $\hat{\theta}$  que minimiza a soma dos quadrados entre os valores observados de  $Y$  e o vetor  $X\hat{\theta}$ :

$$\min(\theta) = \|Y - X\hat{\theta}\|^2 \quad (2.9)$$

Para solucionar a equação 2.9, definimos  $J$  como a função custo onde será calculado o vetor  $\hat{\theta}$  para minimizar  $J$ , que é dado por:

$$J = E_{y_1}^2 + \dots + E_{y_k}^2 = E_y^T E_y \quad (2.10)$$

Substituindo a equação 2.8 na equação 2.10, podemos reescrever da seguinte maneira:

$$J = (Y - X\hat{\theta})^T (Y - X\hat{\theta}) \quad (2.11)$$

$$J = Y^T Y - \hat{\theta}^T (X\hat{\theta})^T Y - Y^T X\hat{\theta} + \hat{\theta}^T (X\hat{\theta}^T) X\hat{\theta} \quad (2.12)$$

Para minimizar a função custo  $J$  com respeito a  $\hat{\theta}$  é necessário resolver  $\frac{\partial J}{\partial \hat{\theta}} = 0$ , (AGUIRRE, 2015), deste modo:

$$\frac{\partial J}{\partial \hat{\theta}} = -X^T y - X^T y + 2X^T X\hat{\theta} \quad (2.13)$$

Igualando a equação 2.13 à 0 tem-se:

$$\hat{\theta} = [X^T X]^{-1} X^T Y \quad (2.14)$$

### 2.1.5 Mínimos Quadrados Recursivos

Os mínimos quadrados recursivos (*Recursive Least Squares*, RLS) gera o mesmo resultado que o *Ordinary Least Squares* (OLS), o qual encontra a curva que melhor se ajusta aos dados minimizando a função custo  $J$ . Porém, no caso do RLS, cada nova amostra atualiza de forma recursiva os parâmetros de ajuste sem a necessidade de usar todo o banco de dados coletados de uma só vez (ROWELL, 2006). Considere o vetor  $\hat{\theta}$  de incógnitas das estimativas de saídas dado por:

$$\hat{\theta} = (X^T X)^{-1} X^T Y \quad (2.15)$$

Onde os vetores  $X$  e  $Y$  são da forma:

$$X(t) = (x(1), x(2), \dots, x(t))^T, \quad Y(t) = (y(1), y(2), \dots, y(t))^T.$$

Com base nessa notação, podemos definir as atualizações dos vetores  $X$  e  $Y$  para o instante  $t + 1$  utilizando as expressões:

$$\begin{cases} X(t+1) = (X(t), x(t+1))^T \\ Y(t+1) = (Y(t), y(t+1))^T \end{cases}$$

Considerando as estimativas através do uso do método dos Mínimos quadrados Batelada, nosso vetor no instante  $t + 1$  assume a forma:

$$\hat{\theta}(t+1) = (X(t+1)^T X(t+1))^{-1} X(t+1)^T Y(t+1) \quad (2.16)$$

Segue ainda da definição do produto entre matrizes que:

$$X(t+1)^T X(t+1) = x(t)^T x(t) + x(t+1)^T x(t+1) \quad (2.17)$$

Tratando do mesmo modo a função  $X(t+1)^T Y(t+1)$  obtemos a igualdade:

$$X(t+1)^T Y(t+1) = x(t)^T y(t) + x(t+1)^T y(t+1) \quad (2.18)$$

Tomando  $P = (X^T X)^{-1}$  e  $R = X^T Y$  podemos reescrever a Equação 2.15 como:

$$\hat{\theta}(t) = P(t)R(t) \quad (2.19)$$

Observe que fazendo essa substituição na equação obtemos:

$$P^{-1}(t+1) = P^{-1}(t) + x(t+1)x(t+1)^T \quad (2.20)$$

$$R(t+1) = R(t) + x(t+1)y(t+1) \quad (2.21)$$



Vamos mostrar que as equações acima são suficientes para efetuar a atualização dos valores das funções  $P$  e  $R$  utilizando apenas o custo computacional de armazenar o valor dessas funções e eliminando a necessidade de efetuar o cálculo da inversa matricial para todos os instantes  $t > 1$ .

A atualização de  $R(t+1)$  é uma consequência da fórmula (2.21) uma vez dispoendo dos valores de  $R(t)$ ,  $x(t+1)$ ,  $y(t+1)$ , de modo que é suficiente analisar o caso  $P(t+1)$ . Pela discussão acima, sabemos que:

$$x(t+1)^T x(t+1) = x(t)^T x(t) + x(t+1)^T x(t) \implies P^{-1}(t+1) = P^{-1}(t) + x(t+1)^T x(t+1).$$

Faremos ainda uso do lema a seguir:

*Lema* (Fórmula da inversão da soma de matrizes): Sejam  $A, B, C, D$  matrizes, quando existe, a inversa de  $(A + BCD)$  pode ser calculada pela fórmula:  $(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}$ . Fazendo a substituição  $A = I$ ,  $B = x_k$ ,  $C = I$ ,  $D = x^t$  obtemos a seguinte equação:

$$P(t+1) = P(t) - P(t)x(t)(I + x(t)^T P(t)x(t))^{-1}x(t)^T P(t) \quad (2.22)$$

Sabendo que nossos valores de saída são escalares, a equação acima pode ser reescrita como:

$$P(t+1) = P(t) - \frac{P(t)x(t)x(t)^T P(t)}{1 + x(t)^T P(t)x(t)} \quad (2.23)$$

Na modelagem do problema considerado, a equação que expressa o erro é dada por  $e(t+1) = y(t+1) - x(t+1)^T \hat{\theta}(t)$  de modo que podemos reescrever o valor de  $R(t+1)$  nos seguintes termos:

$$R(t+1) = R(t) + x(t+1)e(t+1) + x(t+1)x(t+1)^T \hat{\theta}(t) \quad (2.24)$$

Substituindo as equações 2.17 e 2.19 na equação 2.24 obtemos:

$$P^{-1}(t+1)\hat{\theta}(t+1) = P^{-1}(t)\hat{\theta}(t) + x(t+1)e(t+1) + (P^{-1}(t+1) - P^{-1}(t))\hat{\theta}(t) \quad (2.25)$$

Multiplicando a esquerda ambos os membros da equação 2.23 por  $x(t+1)$  e calculando o m.m.c obtemos:

$$P(t+1)x(t+1) = \frac{P(t)x(t+1)}{1 + x(t+1)^T P(t)x(t+1)} \quad (2.26)$$

Denotaremos por  $K(t+1)$  o produto  $P(t+1)x(t+1)$ , de modo que podemos determinar  $\hat{\theta}$  pela equação  $\hat{\theta}(t+1) = \hat{\theta}(t) + K(t+1)e(t+1)$ .

Por fim, condensamos estes resultados em três equações fundamentais:

$$K(t) = \frac{P(t-1)x(t)}{1+x(t)^T P(t-1)x(t)} \quad (2.27)$$

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)(y(t) - x(t)^T \hat{\theta}(t-1)) \quad (2.28)$$

$$P(t) = P(t-1) - K(t)x(t)^T P(t-1) \quad (2.29)$$

Essas três equações compõe o algoritmo de *Recursive Least Squares* (RLS). Para que esse algoritmo seja viabilizado, deve-se inicializar  $\hat{\theta}(t)$  e  $P(t)$  e para isso, define-se valores para  $\hat{\theta}(0)$  e  $P(0)$ .

## 2.2 SINAIS DE ENTRADA

Os sinais de entrada são parte fundamental para a identificação de um modelo. Um sinal de entrada indevidamente escolhido para excitar um sistema pode não revelar o comportamento real em determinadas faixas de operação ou até determinar comportamentos não condizentes com a realidade desse modelo (AGUIRRE, 2015).

Ainda segundo Aguirre (2015), o sinal de excitação da dinâmica de uma planta deve ser persistentemente excitante e possuir ordem suficientemente alta, para excitar um número elevado de frequências, assim resultando um amplo espectro de potência na faixa de frequências desejada. A amplitude da excitação também é importante e deve ser analisada. Um sinal com amplitude muito baixa ou muito alta dará resultados fora da faixa de operação do sistema escolhido, podendo identificar regiões não desejadas, como não-linearidades, ou apresentando uma relação sinal/ruído baixa.

De acordo com Aguirre (2015) os sinais usados para excitar sistema dinâmicos devem possuir as seguintes características:

- Sinais persistentemente excitantes;
- Em sistemas de múltiplas entradas, elas não devem estar correlacionados;
- A largura do espectro do sinal de entrada é importante;

Dentre estas, a mais importante é ser um sinal persistentemente excitante, que ele define sendo uma forma de quantificar quão ativo é um sinal e conseqüentemente quão adequado ele é para a identificação de um sistema. Um sinal persistentemente excitante de ordem  $n$  é um sinal de potência espectral em “ $n$ ” ou mais frequências distintas.

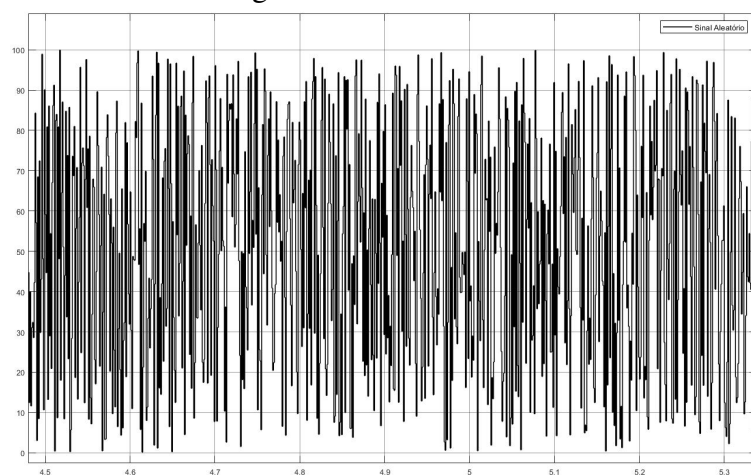
Sinais aleatórios e pseudoaleatórios são comumente usados e atendem os requisitos acima. A seguir, estes e alguns outros sinais utilizados durante este trabalho são brevemente

descritos.

### 2.2.1 Sinais Aleatórios

Um sinal aleatório, ou não determinístico, é um sinal cuja história temporal não pode ser predita (antecipada) de forma exata. A abordagem dos sinais aleatórios se dá pela teoria da probabilidade, em especial através dos conceitos de variáveis aleatórias e processos estocásticos. Na figura 1, pode-se observar um exemplo de sinal aleatório utilizado durante este trabalho.

Figura 1 – Sinal Aleatório



Fonte: O Autor

### 2.2.2 Sequência Binária Pseudoaleatória

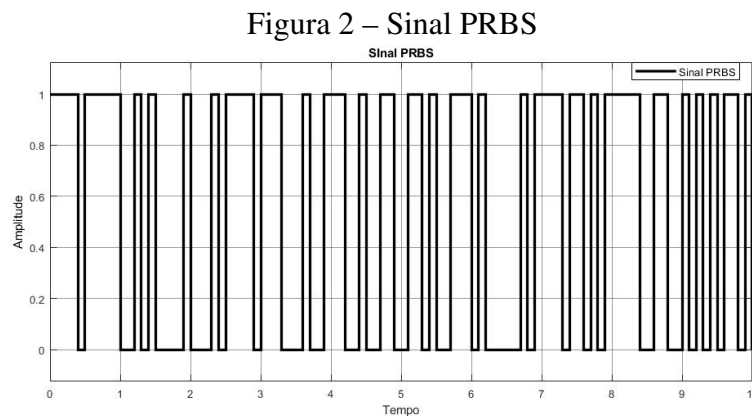
A sequência binária pseudoaleatória (*Pseudo Random Binary Sequence*, PRBS) é o tipo mais comum de sinal pseudoaleatório devido ao seu simples funcionamento. Os sinais *Pseudo Random Binary Sequence* (PRBS) são gerados e caracterizados por um sinal determinístico e periódico de dois níveis ( $+V$  e  $-V$ ). No entanto, este sinal permanece por tempos diferentes nos dois determinados níveis. O modo como esse tempo é determinado favorece a perturbação do sistema por uma ampla faixa do espectro de frequências, tornando esse sinal amplamente usado para os mais variados tipos de sistemas dinâmicos, inclusive alguns sistemas não-lineares. Segundo Aguirre (2015), para gerar sinais adequados e que possam ser usados com sucesso, é importante escolher adequadamente as quatro variáveis envolvidas para geração de um PRBS de modo a não comprometer o funcionamento adequado do sistema, estando dentro de uma faixa de operação desejada.  $n$  é o número de bits que determina a

periodicidade do sinal gerado, e essa mesma periodicidade depende também de  $T_b$ , que é o intervalo entre bits. A periodicidade não deve ser menor que o tempo de acomodação do sistema que está sendo testado.

Se  $T_b$  for muito grande, o sistema interpretará o sinal PRBS como sendo um degrau que resultará em resultados abaixo do esperado. E se  $T_b$  for muito curto, o sistema não terá tempo de resposta para as transições de regimes. Como forma de obter  $T_b$  adequadamente, usa-se a seguinte regra:

$$\frac{t_{min}}{10} < T_b < \frac{t_{min}}{3} \quad (2.30)$$

Sendo  $t_{min}$  a menor constante de tempo de interesse para o sistema dinâmico em questão. Na figura 2, pode-se observar um exemplo de PRBS utilizado durante este trabalho.

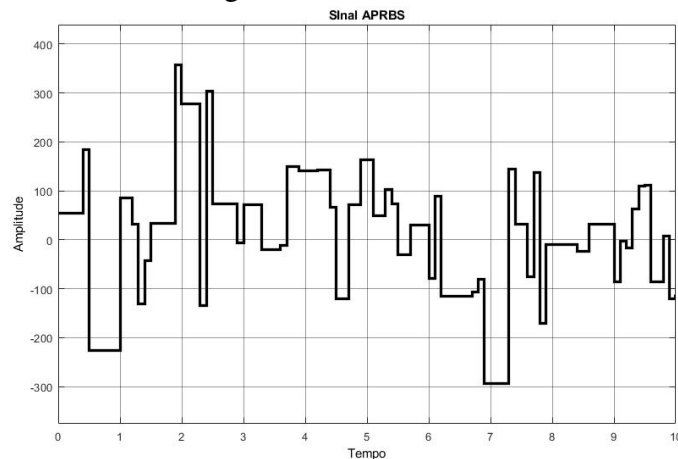


Fonte: O Autor

### 2.2.3 Sequência Binária Pseudoaleatória Modulada em Amplitude

A sequência binária pseudoaleatória de amplitude modulada (*Amplitude Pseudo Random Binary Sequence, APRBS*) foi especialmente importante para a qualidade da identificação do sistema. Sendo uma derivação do sinal PRBS, o *Amplitude Modulated Pseudo Random Binary Sequence (APRBS)* tem as mesmas características destes, porém a amplitude do sinal também muda. O sinal PRBS é utilizado para obtenção de dados para aplicações de identificação de sistemas, entretanto por se um sinal binário, o sinal PRBS não é recomendado para obtenção de dados de sistema não-lineares para identificação de sistemas, sendo necessário o uso do sinal APRBS para identificar comportamento não-lineares (ISERMANN; MÜNCHHOF, 2010). Na figura 8 pode-se observar um exemplo de APRBS utilizado durante este trabalho.

Figura 3 – Sinal APRBS



Fonte: O Autor

## 2.3 REDES NEURAIS ARTIFICIAIS

### 2.3.1 Conceitos Básicos

De acordo com Silva *et al.* (2016), as RNA são modelos computacionais baseados em alguns aspectos do sistema nervoso dos seres humano. Essas redes simulam comportamentos de aquisição de conhecimento e manutenção do conhecimento que podem ser caracterizadas por neurônios artificiais que são representados matematicamente por vetores ou matrizes de pesos sinápticos.

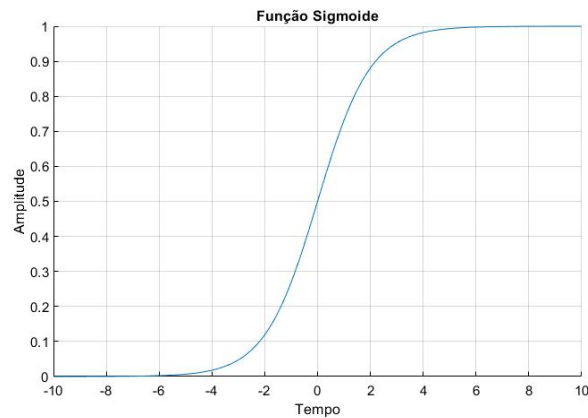
Estes neurônios artificiais são modelos simplificados dos neurônios biológicos, são não-lineares, realizam algumas funções tais como coletar os sinais de entrada, produzir uma resposta de acordo com alguns outros parâmetros como sua função de ativação e pesos sinápticos.

#### 2.3.1.1 Função de ativação

As funções de ativação têm como principal finalidade captar relações não-lineares entre as operações das RNA, principalmente nas camadas ocultas desta rede neural. Vários são os tipos de funções que fazem esse papel. As funções sigmóide e tangente hiperbólica são algum dos exemplos de funções de ativação. Estas estão representadas nas figuras 4 e 5, e suas expressões estão descritas, respectivamente, nas equações 2.32 e 2.31.

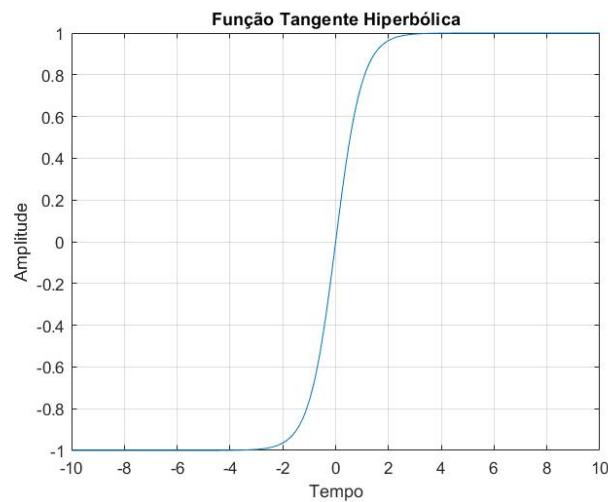
$$f_{sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (2.31)$$

Figura 4 – Função Sigmóide



Fonte: O Autor

Figura 5 – Função Tangente Hiperbólica



Fonte: O Autor

$$f_{tanh}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (2.32)$$

### 2.3.2 *Perceptron Multicamadas*

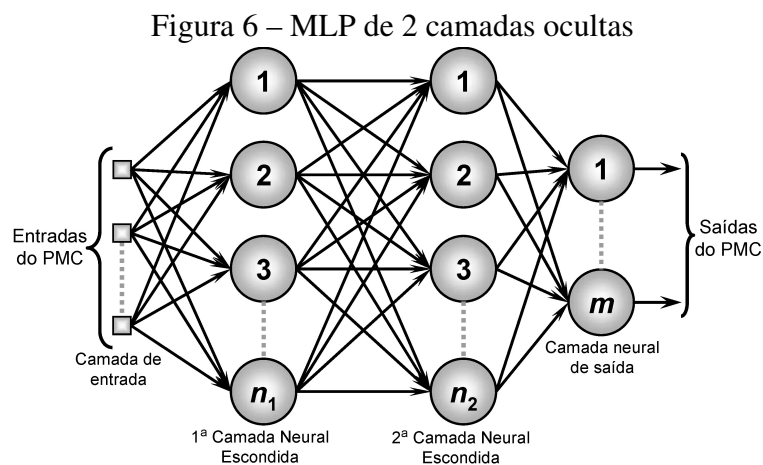
A rede *Perceptron Multicamadas* (*Multi-Layer Perceptron*, MLP) é um tipo de rede neural artificial com alimentação direta (*feedforward*) que consiste em várias camadas de neurônios interconectados. Cada camada é composta de um conjunto de neurônios que recebem entradas e geram saídas. A saída de uma camada é usada como entrada para a próxima camada. A MLP é chamada de *Perceptron* porque é baseada no modelo de neurônio artificial (ROSENBLATT, 1958). Esta foi introduzida em 1974 (WERBOS; JOHN, 1974), e se tornou uma das arquiteturas de rede neural mais populares em aplicações práticas.

Segundo Silva *et al.* (2016), as MLP's possuem uma imensa gama de aplicações com as mais diferentes áreas de conhecimento pela sua arquitetura versátil quanto a sua aplicabilidade:

- Aproximação de funções;
- Reconhecimento de padrões;
- Identificação e controle de sistemas;
- Otimização de sistemas.

### 2.3.2.1 Princípios de Funcionamento

De acordo com Silva *et al.* (2016), no funcionamento básico de uma rede neural artificial, cada entrada da rede neural será propagada uma a uma para a camada de saída. Porém, no caso da MLP que existem uma ou mais camadas intermediárias, as saídas da primeira camada serão as entradas da camada intermediária seguinte. Essa camada intermediária que caracteriza a MLP pode ser composta também de mais de um neurônio, sendo que cada um deles representaria uma das saídas do processo (SILVA *et al.*, 2016) Essa arquitetura permite maior extração das informações referente ao comportamento do sistema e as codificam através de pesos sinápticos e limiares dos seus neurônios, formando uma representação mais fidedigna do sistema tratado. Outra característica dessa arquitetura é que se o processo consiste em  $N$  saídas, a rede neural teria  $N$  neurônios em sua última camada. Na figura 6, pode-se observar o modelo de uma rede MLP com duas camadas ocultas.



Fonte: (SILVA *et al.*, 2016)

### 2.3.3 Treinamento de uma MLP

De acordo com Silva *et al.* (2016), o treinamento de uma MLP é feito por um algoritmo chamado BP, que é um algoritmo supervisionado que usa a técnica de gradiente descendente para ajustar os pesos sinápticos e limiares da rede neural para minimizar o erro entre os valores de saída das camadas da MLP e o valor desejado. O algoritmo acontece em duas etapas: a *Foward Propagation* e a *Backward Propagation*. A primeira fase acontece quando os sinais de entrada da rede neural  $\{x_1, x_2, \dots, x_n\}$  são inseridos na entrada da rede e são propagados camada a camada até a produção das respectivas saídas. Resumidamente, essa etapa visa apenas obter as respostas da rede levando em consideração os valores dos pesos sinápticos e limiares que foram iniciados com pequenos valores aleatórios. Após isso, como se trata de um processo de aprendizagem supervisionado, os valores das saídas serão comparados e os erros em relação à resposta desejada serão calculados. E esses erros serão usados para atualizar os pesos e limiares de todos os neurônios.

O algoritmo de treinamento BP será descrito a seguir. A simplificação da MLP é para facilitar a demonstração matemática e é uma adaptação de (SILVA *et al.*, 2016). Supondo que temos uma MLP com 1 camada oculta, a camada de entrada e a camada de saída.  $x$  é um vetor unitário de entrada,  $y$  um vetor unitário de saída,  $w$  o peso da camada oculta,  $v$  o peso da camada de saída e  $f$  a função de ativação.

O algoritmo de treinamento *backpropagation* funciona com os seguintes passos:

1. Inicialização do peso:

- O peso pode ser inicializado aleatoriamente ou com algum valor específico dependendo do método de inicialização;

2. Etapa de propagação (*Foward Propagation*):

- Calcule a saída da camada oculta:
  - a) Calcule a entrada líquida *net* para cada neurônio da camada oculta:

$$net_h = (w * x), \quad (2.33)$$

onde  $w$  é o peso da camada oculta e  $x$  é valor de entrada.

- b) Aplica-se a função de ativação na entrada líquida para obter a saída do neurônio:

$$out_h = f(net_h) = f(w * x) \quad (2.34)$$



- Calcula-se a saída da camada de saída:

a) Calcula-se a entrada líquida para cada neurônio da camada de saída:

$$net_o = (v * out_h), \quad (2.35)$$

onde  $v$  é o peso da camada de saída e  $out_h$  é a saída do neurônio da camada oculta.

b) Aplica-se a função de ativação na entrada líquida para obter a saída do neurônio:

$$out_o = f(net_o) = (v * out_h) \quad (2.36)$$

3. Cálculo do erro:

- Calcula-se o erro quadrático entre a saída esperada ( $\hat{y}$ ) e a saída da MLP:

$$E = \frac{1}{2}(\hat{y} - y)^2, \quad (2.37)$$

onde  $E$  é o erro quadrático.

4. Etapa de retropropagação (*backpropagation*):

- Calcula-se o gradiente do erro em relação aos pesos da camada de saída;

$$\delta_o = (\hat{y} - out_o) * f'(net_o), \quad (2.38)$$

onde  $f'$  é a derivada da função de ativação.

- Atualiza-se os pesos da camada de saída:

$$v_{novo} = v + LR * \delta_o * out_h, \quad (2.39)$$

onde  $LR$  é a taxa de aprendizagem.

- Calcula-se o gradiente do erro em relação ao peso da camada oculta:

$$\delta_h = f'(net_h) * \frac{\delta_o * v_o}{\delta_o + v_o} \quad (2.40)$$

- Atualiza-se os pesos da camada oculta:

$$W_{novo} = W + LR * \delta_h * x, \quad (2.41)$$

Repete-se a etapas 2 e 4 até que um critério de parada seja atingido, como um número máximo de iterações ou um erro mínimo.

### 2.3.4 Máquina de Aprendizado Extremo

A máquina de aprendizado extremo (*Extreme Machine Learning*, ELM) é um algoritmo de aprendizado de máquina que pertence à família de RNA com alimentação direta (*feedforward*). Esta foi desenvolvida como uma alternativa mais rápida e eficiente aos métodos tradicionais de treinamento de redes neurais (HUANG *et al.*, 2011). A principal diferença entre o ELM e outros algoritmos de treinamento de RNA é que o ELM usa uma abordagem de treinamento de uma única etapa. A rede ELM tem uma estrutura similar a rede MLP, e é formada apenas por uma camada de entrada e uma camada oculta de neurônios. Os pesos da camada de entrada são definidos de forma aleatória, e os pesos das conexões entre a camada oculta e a saída são atualizados utilizando a técnica dos mínimos quadrados.

Neste trabalho, além dos mínimos quadrados ordinários, utilizou-se a técnica dos mínimos quadrados recursivos para calcular os pesos da camada oculta da rede ELM. Estes dois treinamentos serão diferenciados pelos termos *Extreme Learning Machine Batch* ( $ELM_b$ ) e *Extreme Learning Machine Online* ( $ELM_o$ ).

#### 2.3.4.1 ELM Não Recursivo

Para  $N$  amostras arbitrárias e distintas  $(x_i, y_i)$ , onde  $x_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T$  e  $y_i = [y_{i1}, y_{i2}, \dots, y_{in}]^T$ , uma ELM padrão com neurônios na camada oculta e  $g(x)$  a função de ativação, pode ser modelada como (HUANG *et al.*, 2004):

$$y_j = \sum_{i=1}^N \beta_i g(w_i x_j + b_i) = o_j, j = 1, \dots, N, \quad (2.42)$$

onde  $w_i = [w_{i1}, w_{i2}, \dots, w_{in}]^T$  é o vetor de peso conectando o  $i$ -ésimo neurônio da camada oculta aos neurônios de entrada,  $\beta_i = [\beta_{i1}, \beta_{i2}, \dots, \beta_{im}]^T$  é o vetor de peso conectando o  $i$ -ésimo neurônio oculto aos neurônios de saída e  $b_i$  é o limiar do  $i$ -ésimo neurônio da camada oculta.  $w_i x_j$  denota o produto interno de  $w_i$  e  $x_j$ . No caso de tarefas de regressão, os neurônios da camada de saída possuem função de ativação linear.

Com a ELM nas condições iniciais, o objetivo da  $ELM_b$  é aproximar de 0 a diferença entre os valores reais ( $y_j$ ) dos valores dos neurônios da camada de saída ( $o_j$ ).

$$\sum_{j=1}^N ||o_j - y_j|| = 0, \quad (2.43)$$

Para que a condição da equação 2.43 seja verdadeira, existem  $\beta_i$ ,  $w_i$  e  $b_i$  tais que:

$$\sum_{i=1}^N \beta_i g(w_i x_j + b_i) = y_j, j = 1, \dots, N, \quad (2.44)$$

As  $N$  equações podem ser escritas matricialmente da seguinte maneira:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \quad (2.45)$$

onde:

$$\mathbf{H}(w_1, \dots, w_{\tilde{N}}, b_1, \dots, b_{\tilde{N}}, x_1, \dots, x_n) = \begin{bmatrix} g(w_1 x_1 + b_1) & \dots & g(w_{\tilde{N}} x_1 + b_{\tilde{N}}) \\ \vdots & \dots & \vdots \\ g(w_1 x_N + b_1) & \dots & g(w_{\tilde{N}} x_N + b_{\tilde{N}}) \end{bmatrix} \quad (2.46)$$

$$\boldsymbol{\beta} = [\beta_1^T \dots \beta_{\tilde{N}}^T] \quad (2.47)$$

e

$$\mathbf{T} = [t_1^T \dots t_N^T] \quad (2.48)$$

$\mathbf{H}$  é a matriz de saída da camada oculta da rede neural; a  $i$ -ésima coluna de  $\mathbf{H}$  é o  $i$ -ésimo vetor de saída dos neurônios da camada oculta em relação as entradas  $x_1, x_2, \dots, x_N$ . Usando o algoritmo do OLS descrito na seção 2.1.4, a solução da equação 2.45 é dada por:

$$\hat{\boldsymbol{\beta}} = [(\mathbf{H}^T \mathbf{H})]^{-1} \mathbf{H}^T \mathbf{T} \quad (2.49)$$

Com essas soluções chega-se ao algoritmo de treinamento de uma ELM que é definida a partir dos seguintes passos:

1. Atribuir um peso inicial  $w_i$  e um *bias*  $b_i$  arbitrariamente;
2. Calcular a matriz de saída da camada oculta  $\mathbf{H}$ ;
3. Calcular o peso de saída  $\boldsymbol{\beta}$  definida pela equação a seguir:

$$\hat{\boldsymbol{\beta}} = [(\mathbf{H}^T \mathbf{H})]^{-1} \mathbf{H}^T \mathbf{T} \quad (2.50)$$

onde  $\mathbf{H}$ ,  $\boldsymbol{\beta}$  e  $\mathbf{T}$  são definidos nas equações 2.46, 2.47 e 2.48.

### 2.3.4.2 ELM Recursivo

De acordo com Wang e Lu (2021), e seguindo as mesmas condições iniciais do  $ELM_b$ , podemos adicionar o conceito de inversa a esquerda que será útil mais à frente. Dada a equação 2.49, podemos simplificar da seguinte maneira:

$$\beta = \mathbf{H}^\dagger \mathbf{T} \quad (2.51)$$

Onde  $\mathbf{H}^\dagger$  é a matriz pseudo inversa da saída da camada oculta, descrita matematicamente a seguir:

$$\mathbf{H}^\dagger = [\mathbf{H}^T \mathbf{H}]^{-1} \mathbf{H}^T \mathbf{T} \quad (2.52)$$

Dado uma matriz  $\mathbf{A}$ , uma matriz  $\mathbf{B}$  que satisfaz  $\mathbf{BA}=\mathbf{I}$  é chamada de inversa à esquerda de  $\mathbf{A}$ . A partir desse conceito a matriz  $\mathbf{H}^\dagger$  é uma matriz pseudo inversa a esquerda. Substituindo a equação 2.52 na equação 2.51 temos:

$$\beta = [(\mathbf{H}^T \mathbf{H})]^{-1} \mathbf{H}^T \mathbf{T} \quad (2.53)$$

Para alcançar o algoritmo de recursividade, é necessário definir um valor de  $\beta$  inicial ( $\beta_0$ ) e definir uma variável auxiliar  $\mathbf{M}$  também com um valor inicial ( $\mathbf{M}_0$ ) (HUANG *et al.*, 2005):

$$\beta_0 = \mathbf{M}_0 \mathbf{H}_0^T \mathbf{T}_0 \quad (2.54)$$

onde,

$$\mathbf{M}_0 = \mathbf{H}_0^T \mathbf{H}_0^{-1} \quad (2.55)$$

Para o algoritmo do  $ELM_o$  define-se  $\mathbf{S} = [(x_i, y_i) \in R^n, R^m, i = 1, \dots, N]$  como o conjunto de treinamento (WANG *et al.*, 2022):

1. Inicialização:  $k = 0$ ,
2. Calcula-se a matriz de saída da camada oculta  $H_0$  usando o conjunto de treinamento e os parâmetros  $w_i$  e *bias*  $b_i$  aleatoriamente da mesma forma do método RLS;
3. Obtêm-se os pesos de saída: usando as equações 2.54.
4. **Sequência de treinamento *online***: Para cada novo bloco de dados  $[x_i, y_i]$ , atualize a saída da camada oculta  $\mathbf{H}_{(k+1)} = [h_k, h_{(k+1)}]$ , onde  $h_{(k+1)}$  é a saída da camada oculta do novo bloco de dados. Atualize  $\beta_{k+1}$  a partir das seguintes equações:

$$\beta_{(k+1)} = \beta_t + \mathbf{M}_{(k+1)} h_{(k+1)} (t_i^T - h_{(k+1)})^T \beta_{(k)}, \quad (2.56)$$

$$\mathbf{M}_{(k+1)} = \mathbf{M}_k - \frac{\mathbf{M}_k h_{(k+1)} h_{(k+1)}^T \mathbf{M}_k}{1 + h_{(k+1)}^T \mathbf{M}_k h_{(k+1)}} \quad (2.57)$$

## 2.4 Validação do Modelo

Após passar pelas diversas etapas que compõe a identificação de sistema chega-se ao produto final, um modelo. Em problemas de validação, a questão crucial é saber se o modelo encontrado é válido ou não. O tipo de validação depende de qual o uso pretendido para o modelo encontrado. Comparar a simulação do modelo obtido com dados medidos é provavelmente a forma mais usual de se validar um modelo. Nesse caso deseja-se saber se o modelo reproduz ao longo do tempo os dados observados (AGUIRRE, 2015).

Um cuidado básico é não usar os dados utilizados para obter o modelo na validação. Como deseja-se saber o quão geral é o modelo, é necessário usar um outro conjunto de dados para fazer a validação (AGUIRRE, 2015).

Dois tipos de validação são coerentes para o tipo de modelo encontrado. A validação por *1-passo à frente* e a validação por *simulação livre*.

A validação com um passo à frente envolve a utilização de medições e parâmetros estimados para fazer previsões para o passo de tempo seguinte, repetindo este processo para os passos de tempo subsequentes.

Supondo  $\hat{y}$  o valor da predição e um vetor de regressores com 2 amostras atrasadas de entrada e 2 amostras atrasadas de saída, essas amostras são necessárias para inicialização do modelo com os valores medidos (AGUIRRE, 2015), temos o vetor de regressores na equação 2.58.

$$\hat{y}^T(k-1) = [y(k-1)y(k-2)u(k-1)u(k-2)] \quad (2.58)$$

A predição por *1-passo à frente* segue os seguintes passos:

$$\hat{y}(3) = [y(2)y(1)u(2)u(1)]\hat{\theta} \quad (2.59)$$

$$\hat{y}(4) = [y(3)y(2)u(3)u(2)]\hat{\theta} \quad (2.60)$$

$$\hat{y}(5) = [y(4)y(3)u(4)u(3)]\hat{\theta} \quad (2.61)$$

$$\dots \quad (2.62)$$

$$\hat{y}(k) = [y(k-1)y(k-2)u(k-1)u(k-2)]\hat{\theta}, \quad (2.63)$$

Esta abordagem garante que o modelo não tem tempo para cometer erros significativos porque só dá um passo em frente antes de ser corrigido por novos dados. As previsões de *1-passo à frente* não conseguem evidenciar as deficiências do modelo. Isto é útil mesmo para modelos instáveis, porque estes só dão um passo na direção da instabilidade antes de serem corrigidos. Tem aplicações para previsões de tempo e mercado de ações onde as medidas estão disponíveis e aplicadas no curto prazo (AGUIRRE, 2015).

A *simulação livre* é o procedimento em que as previsões e as entradas são utilizadas para inicializar um modelo e estimar parâmetros, permitindo previsões futuras. Inicializando a validação com  $\hat{y}$  sendo o valor da previsão e com o mesmo vetor de regressores da equação 2.58:

$$\hat{y}(3) = [y(2)y(1)u(2)u(1)]\hat{\theta} \quad (2.64)$$

$$\hat{y}(4) = [\hat{y}(3)y(2)u(3)u(2)]\hat{\theta} \quad (2.65)$$

$$\hat{y}(5) = [\hat{y}(4)\hat{y}(3)u(4)u(3)]\hat{\theta} \quad (2.66)$$

$$\dots \quad (2.67)$$

$$\hat{y}(k) = [\hat{y}(k-1)\hat{y}(k-2)u(k-1)u(k-2)]\hat{\theta}, \quad (2.68)$$

Esse procedimento é, ao contrário da previsão 1-passo à frente, uma boa maneira de testar se o modelo consegue explicar as observações realizadas. Este também é usado quando é necessário substituir o sistema real pelo modelo identificado (AGUIRRE, 2015).

#### 2.4.1 Medida de Erro - Erro Quadrático Médio

O Mean Squared Error (MSE) é uma medida comumente usada para avaliar a qualidade de um modelo de previsão ou regressão. Ele é calculado pela média da soma do quadrado das diferenças entre os valores reais e os valores preditos pelo modelo. É descrito genericamente a seguir:

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (2.69)$$

Onde  $y_i$  é o valor da saída real e  $\hat{y}_i$  o valor da saída predita.

O MSE é uma medida de erro muito utilizada se o modelo obtido tem como finalidade de realizar previsões.

### 3 MATERIAIS E MÉTODOS

#### 3.1 Sistema Utilizado

O modelo de um motor CC foi utilizado como base para a geração de dados através de uma simulação no software MATLAB (*Simulink*). A escolha por este modelo foi feita em razão do conhecimento prévio das equações que relacionam suas entradas e saídas, facilitando a comparação entre os resultados estimados pelos modelos de aprendizado de máquina e os resultados gerados pela simulação. É importante mencionar que os algoritmos aqui testados não se limitam à esse sistema. Como o sistema utilizado foi tratado como "caixa preta", para utilizar estes modelos, basta obter um banco de dados representativo de entradas e saídas de um sistema qualquer.

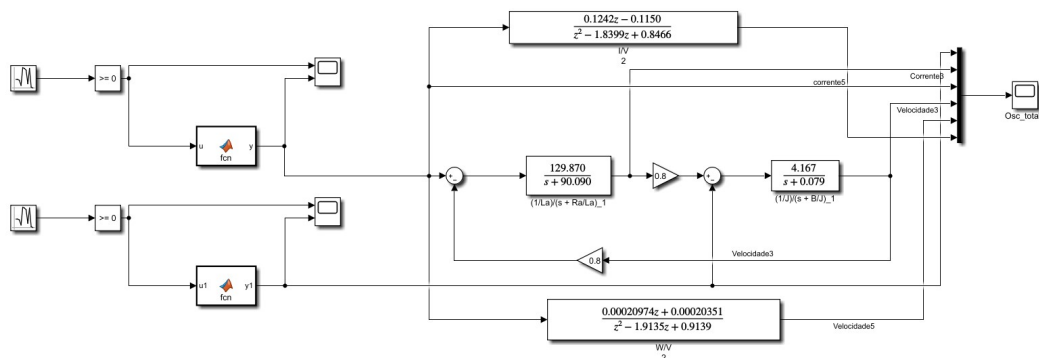
O diagrama de blocos, implementado no ambiente (*Simulink*), que representa o modelo utilizado do motor CC, está ilustrado nas figuras 7 e 8. Primeiramente, como entradas externas ao sistema, foram utilizados os seguintes sinais:

- $V_a$ : Tensão de entrada [V];
- $C_l$ : carga [N m].

Já como sinais de saída, foram considerados os seguintes sinais:

- $I_a$ : corrente [A];
- $W$ : Velocidade do eixo [rad/s].

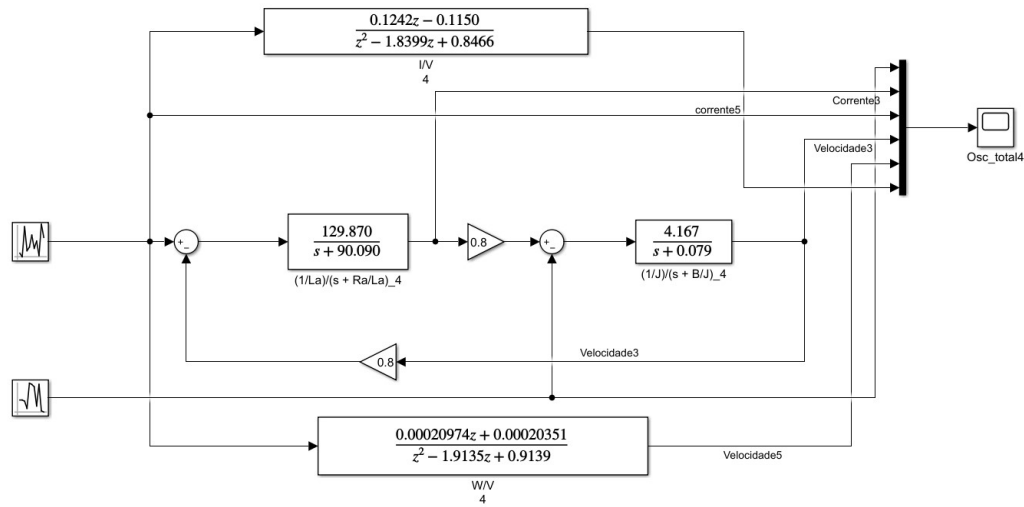
Figura 7 – Diagrama de blocos do motor CC no Simulink com entradas APRBS



Fonte: O Autor

Os blocos gerados são baseados nas equações físicas que regem esses sistemas. Simplificando estes diagramas de blocos, podemos obter as funções de transferência que relacionam todos os sinais envolvidos. Considerando  $C_l = 0$ , temos as seguintes funções de transferência

Figura 8 – Diagrama de blocos do motor CC no Simulink com entradas aleatórias



Fonte: O Autor

envolvendo tensão de entrada, velocidade e corrente de armadura:

$$\frac{w(s)}{V_a(s)} = \frac{\frac{K}{L_a J}}{\left(s + \frac{R_a}{L_a}\right)\left(s + \frac{B}{J}\right) + \frac{K^2}{L_a J}} \quad (3.1)$$

$$\frac{I_a(s)}{V_a(s)} = \frac{\frac{1}{L_a}\left(s + \frac{B}{J}\right)}{\left(s + \frac{R_a}{L_a}\right)\left(s + \frac{B}{J}\right) + \frac{K^2}{L_a J}} \quad (3.2)$$

Onde:

- $J$ : Momento de Inércia do rotor [Kgm<sup>2</sup>];
- $B$ : Coeficiente de atrito [Nms<sup>2</sup>/rad];
- $K$ : Constante de torque [Nm/A];
- $R_a$ : Resistência de armadura [ $\omega$ ];
- $L_a$ : Reatância de armadura [H]

Além disso, considerando  $V_a = 0$ , podemos obter as funções de transferência que relacionam carga aplicada, velocidade e corrente de armadura:

$$\frac{w}{C_l} = \frac{\left(\frac{1}{s}\right)\left(s + \frac{R_a}{L_a}\right)}{\left(s + \frac{R_a}{L_a}\right)\left(s + \frac{B}{J}\right)\frac{K^2}{L_a J}} \quad (3.3)$$

$$\frac{I_a}{C_l} = \frac{\frac{K^2}{L_a J}}{\left(s + \frac{R_a}{L_a}\right)\left(s + \frac{B}{J}\right) + \frac{K^2}{L_a J}}, \quad (3.4)$$



onde os parâmetros  $J$ ,  $B$ ,  $K$ ,  $R_a$ , e  $L_a$  já foram definidos anteriormente, e os valores destes são mostrados na Tabela 1.

De modo a excitar esse sistema com formas de onda de tensão e carga, foram utilizados os sinais descritos no capítulo 2: Aleatório e APRBS. No ambiente de simulação Simulink não existe um modelo padrão para os sinais APRBS. Por isso, os blocos destes sinais foram implementados. O algoritmo destes blocos está descrito no Apêndice A.

### 3.2 Banco de dados

Das simulações geradas a partir dos sinais de entrada e das equações e parâmetros do sistema utilizado, foram construídos dois bancos de dados para iniciar o processo de identificação de sistemas com os modelos de aprendizado de máquinas.

As simulações tiveram uma duração de 10,006 segundos, e uma taxa de amostragem de 1 kHz. No banco de dados 1, os sinais de tensão e carga foram gerados na forma aleatória, e foram medidos 4 sinais ( $V_a$ ,  $I_a$ ,  $C_l$  e  $W$ ). Já para o banco de dados 2, os sinais de tensão e carga foram gerados a partir de um APRBS, e foram medidos os mesmos sinais do banco de dados 1. Deste modo, cada banco de dados é formado de 4 sinais com 10.006 amostras cada.

### 3.3 Organização dos experimentos

Para realização das comparações e geração de resultados há alguns hiperparâmetros que serão fixos para qualquer rede neural utilizada. São estes o número de realizações, a divisão do banco de dados entre treinamento e predição, o tipo de validação do modelo, e os números de amostras atrasadas de entrada/saída. Os valores para esses parâmetros são expostos na tabela 2 e serão detalhados a seguir:

- Número de realizações: é a etapa que compreende a divisão do banco de dados, normalização, treino, teste e geração das estatísticas;
- Número de amostras atrasadas: para treinar modelos de aprendizado de máquina, são

Tabela 1 – Parâmetros do Motor CC

Parâmetros do Motor CC	Valores(unidade)
$J$	0,239981 $Kgm^2$
$B$	0,329193 $Nms^2/rad$
$K$	1 $Nm/A$
$R_a$	0,0077 $\omega$
$L_a$	0,693694 $H$

necessários vários vetores de atributos e vetores de saída desejada. No caso de identificação de sistemas, os vetores de atributos são compostos de amostras atrasadas de sinais de entrada e saída. Já os vetores de saída desejada são compostos de amostras atuais dos sinais de saída;

- Neste trabalho, como o modelo ARX foi o escolhido para identificação de sistemas, foram utilizadas duas amostras atrasadas tanto de entradas como de saídas para formar cada um dos vetores de atributos.
- Amostras utilizadas para treino: de modo a gerar os vetores de atributos e saídas desejadas para o treinamento dos modelos, 50% do total de amostras de cada sinal foram utilizados. Os demais são utilizados para validação;
- Método de validação do modelo: neste trabalho, foi utilizada a simulação livre, onde os valores estimados pelos modelos, de saídas passadas, são utilizados para estimar as saídas atuais.

Tabela 2 – Parâmetros padrão para as simulações

Número de Realizações	10
Amostras utilizadas para treinamento	50%
Método para validação do modelo	Simulação livre
Número de amostras atrasadas de saída(p)	2
Número de amostras atrasadas de entrada(q)	2

### 3.4 Redes neurais – Seleção do Modelo

Nesta seção são explicitados como e quais hiperparâmetros das redes neurais utilizadas nesse trabalho foram configurados para obter os resultados que serão expostos no capítulo 4. É importante mencionar que todas essas redes foram implementadas sem a utilização de nenhuma *toolbox* específica, utilizando apenas programação orientada a objetos na linguagem nativa do software Matlab.

#### 3.4.1 MLP

De modo a construir modelos a partir da rede MLP, os seguintes hiperparâmetros devem ser definidos: número de épocas, números de neurônios na camada oculta, taxa de aprendizagem e a função de ativação. A faixa de valores testados para esse algoritmo está descrita na Tabela 3. É importante salientar que todas as redes neurais utilizadas neste trabalho

possuem apenas uma camada oculta.

Tabela 3 – Hiper parâmetros MLP

Hiper Parâmetros	
Número de épocas	20 a 100
Números de neurônios na camada oculta	2 a 20
Taxa de aprendizagem	0.001 a 0.1
Função de ativação	sigmoide

### 3.4.2 ELM Batelada

Para a construção do modelo  $ELM_b$ , são necessários apenas dois hiperparâmetros: número de neurônios na camada oculta e função de ativação. Os valores destes são descritos na Tabela 4

Tabela 4 – Hiper Parâmetros ELM Não Recursiva

Número de neurônios na camada oculta	25
Função de ativação	sigmoide

### 3.4.3 ELM Online

Para as simulações realizadas com as redes neurais do tipo  $ELM_o$  são necessários a inicialização dos hiperparâmetros próprios das redes neurais  $ELM_o$ . Assim como a MLP, foram definidos intervalo de valores para cada hiperparâmetro que serão descritos na tabela 5:

Tabela 5 – Hiper parâmetros ELM Recursiva

Hiperparâmetros	
Números de neurônios na camada oculta	25 a 50
$W_{inicial}$	0.1 a 0.001
$P_{inicial}$	1.000 a 100.0000
Função de ativação	Sigmoide ou tangente hiperbólica

O  $W_{inicial}$  é definido na subseção 2.3.4.1 como sendo  $w_i$ , o vetor de pesos conectando o  $i$ -ésimo neurônio da camada oculta aos neurônios de entrada. Na subseção 2.3.4.2 é descrito que o  $W_{inicial}$  deve ser iniciado aleatoriamente para iniciar o algoritmo de recursividade da ELM. Já o  $P_{inicial}$  é definido nas equações 2.22 e 2.29 e deve ser iniciado aleatoriamente como descrito na subseção 2.1.5.

## 4 RESULTADOS E DISCUSSÕES

De acordo com os métodos descritos no capítulo 3 deste trabalho, serão indicados os resultados das simulações com os diferentes tipos de redes neurais e hiperparâmetros. Os resultados serão mostrados a partir de cada tipo de entrada como descrito na seção 2.2 e com o banco de dados relacionado a esse tipo de entrada. As simulações serão divididas em dois blocos. No primeiro bloco, são mostrados os resultados com o banco de dados 1 (onde os valores de tensão e carga foram gerados de forma aleatória). Já no segundo bloco, são mostrados os resultados com o banco de dados 2 (onde os valores de tensão e carga foram gerados a partir de sinais APRBS).

### 4.1 Sinal de entrada de tensão e carga Aleatórios

#### 4.1.1 MLP

Como descrito no capítulo 3, os hiperparâmetros da MLP variam em determinados valores de acordo com a tabela 3. Para esse banco de dados de entrada foram realizados apenas dois testes como descritos a seguir.

##### *Teste 01*

Para o teste 01 usaremos os seguintes hiperparâmetros:

- Número de épocas = 60;
- Número de neurônios na camada oculta = 10;
- Função de ativação = sigmoide;
- Taxa de aprendizagem = 0.05.

A partir desses hiperparâmetros temos os seguintes resultados. A Figura 9 e a Figura 10 apresentam curvas sobrepostas dos valores reais do banco de dados e dos dados preditos com a metodologia usada e os hiperparâmetros selecionados. A variável da Figura 9 é a velocidade e a variável de saída da Figura 10 é a corrente.

Figura 9 – Predição velocidade MLP teste 01

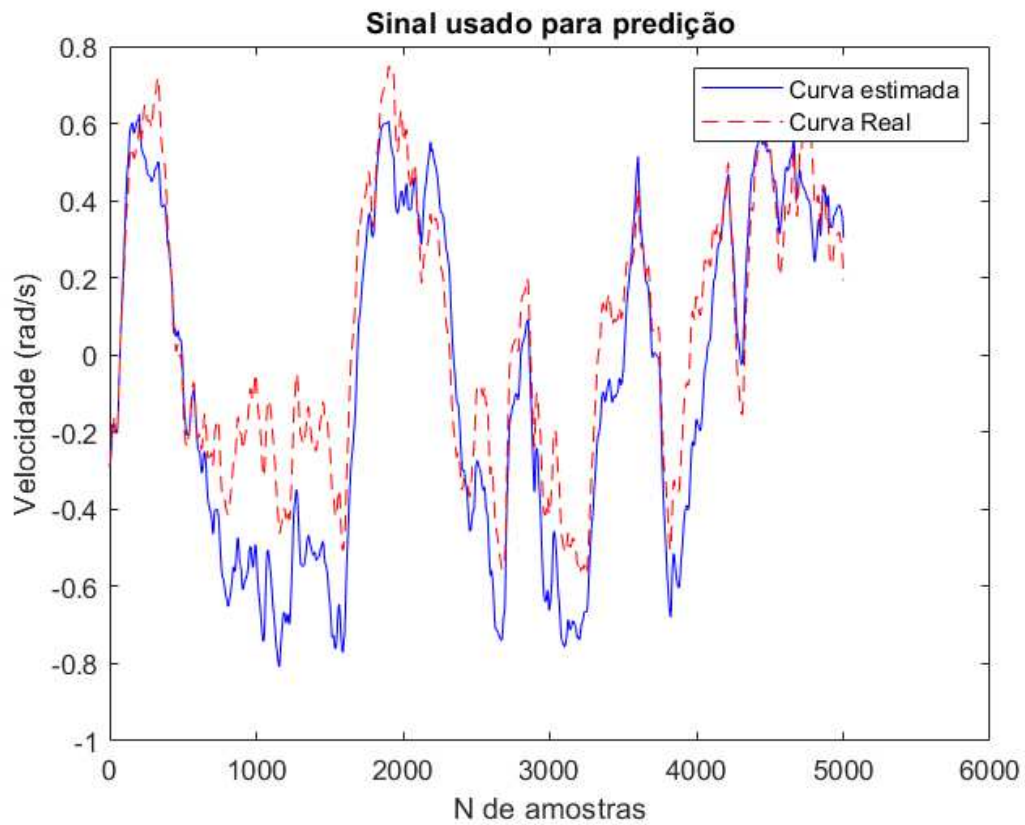
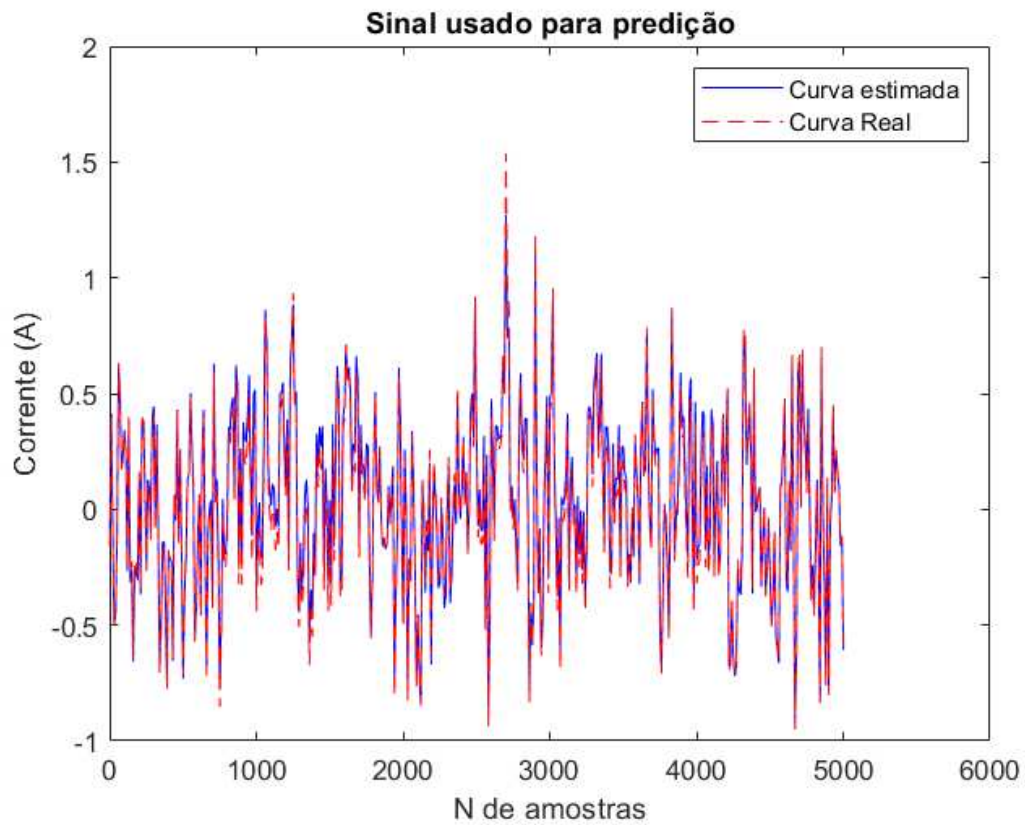


Figura 10 – Predição corrente MLP teste 01



Visualmente, há uma discordância considerável entre as curvas nos valores preditos da velocidade. Nos valores preditos de corrente em relação aos valores reais há uma diferença visível, mas menor em relação a Figura 9. Numericamente, com a medida para quantificar essa discordância, o MSE da velocidade para o teste 01 é de  $1.7390 \times 10^{-2}$  e o MSE para a corrente é de  $5.8418 \times 10^{-3}$ .

### Teste 02

Em comparação ao teste 01, essa simulação apresenta apenas uma mudança no número de neurônios na camada oculta, mantendo os outros hiperparâmetros inalterados.

- Número de épocas = 60;
- Número de neurônios na camada oculta = 20;
- Função de ativação = sigmoide;
- Taxa de aprendizagem = 0.05.

A variável da figura 11 é a velocidade e a variável de saída da figura 12 é a corrente.

Figura 11 – Predição velocidade teste 02

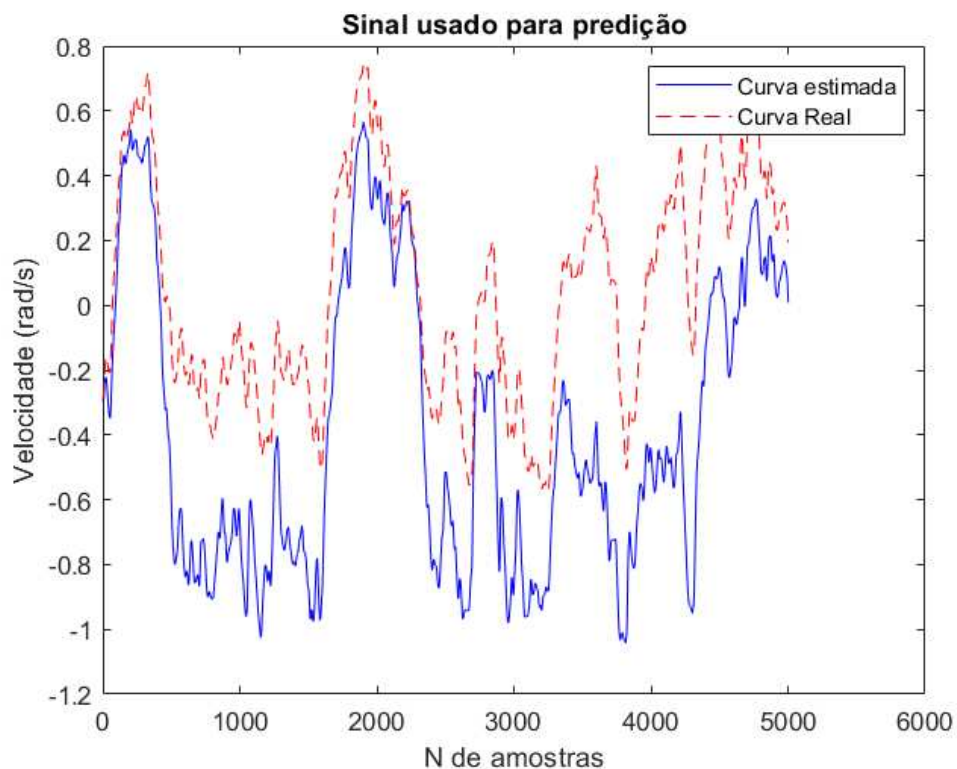
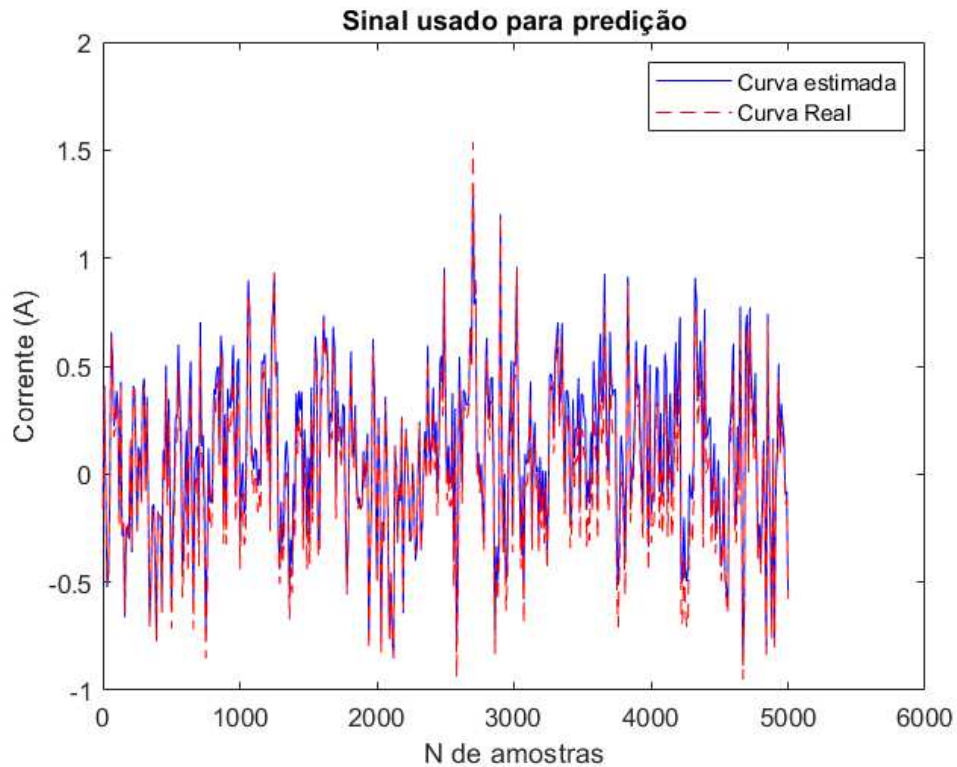


Figura 12 – Predição corrente teste 02



Nesses resultados também se percebe grandes discrepâncias na saída de velocidade quando analisadas visualmente na figura 11. Já para a corrente, as diferenças ocorrem em alguns picos e podem ser observadas. Numericamente, o erro quadrático da velocidade é de  $1.7463 \times 10^{-1}$  e da corrente é de  $1.1483 \times 10^{-2}$ .

A partir dos testes 01 e 02, onde foram usados os mesmos dados de entrada e saída, pode-se fazer observações. A primeira delas é a importância dos hiperparâmetros para o bom funcionamento e a qualidade dos resultados com o menor MSE possível. Há diferenças consideráveis quando se compara a ordem de grandeza dos erros quadráticos médios como mostrado na tabela abaixo:

Tabela 6 – Resumo dos testes MLP com entradas aleatórias

Variável de saída	Teste 01	Teste 02
Velocidade	$1.7390 \times 10^{-2}$	$1.7463 \times 10^{-1}$
Corrente	$5.8418 \times 10^{-3}$	$1.1483 \times 10^{-2}$

Os resultados do teste 01 apresentam menor MSE em comparação ao teste 02. Como forma de simplificar, fixou-se a o número de épocas dos dois testes, e mudou-se apenas o número de neurônios na camada oculta. Neste caso, pode-se perceber que um aumento no número de

neurônios na camada oculta não necessariamente leva a uma melhora de desempenho do modelo.

#### 4.1.2 ELM Não recursivo

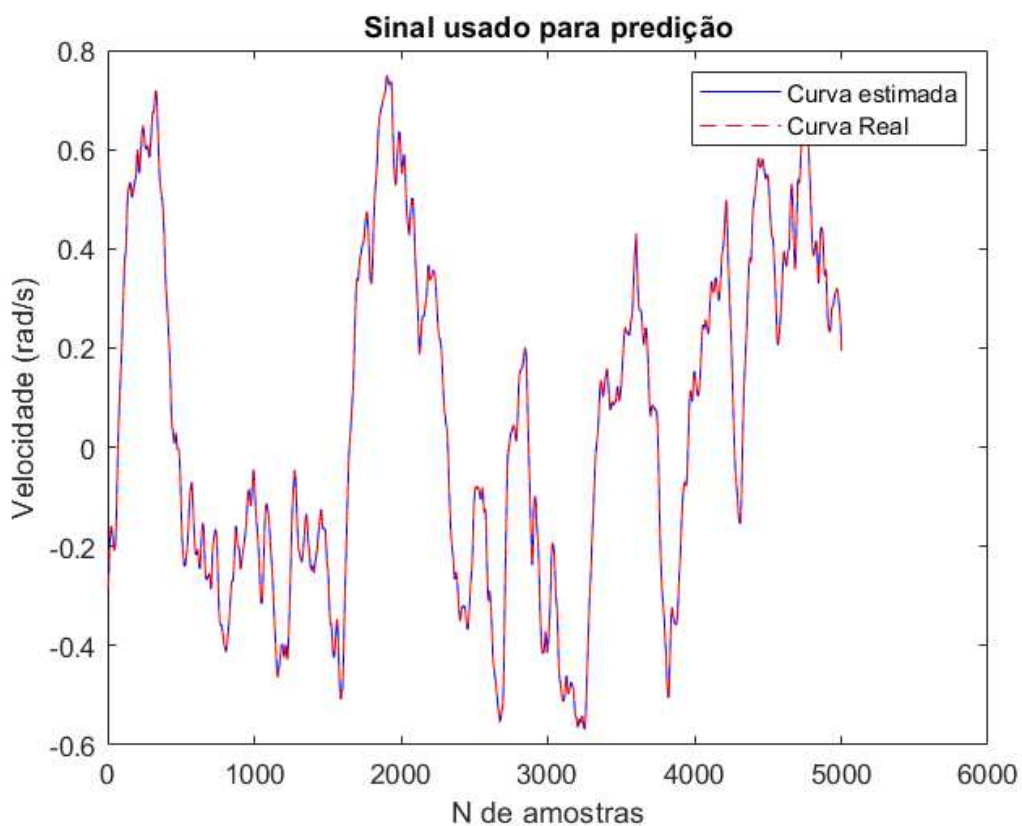
Usaremos agora a rede neural  $ELM_b$ , e para essa rede os hiperparâmetros são os descritos na tabela 4.

##### Teste 03

- Número de neurônios na camada oculta = 50;
- Função de ativação = sigmoide;

Os resultados desta rede neural, aplicados ao banco de dados 1 são ilustrados nas figuras 13 e 14, onde as variáveis estimadas são, respectivamente, a velocidade e a corrente. Os gráficos representam a curva de saída real com a curva de valores preditos de acordo com a legenda presente na figura.

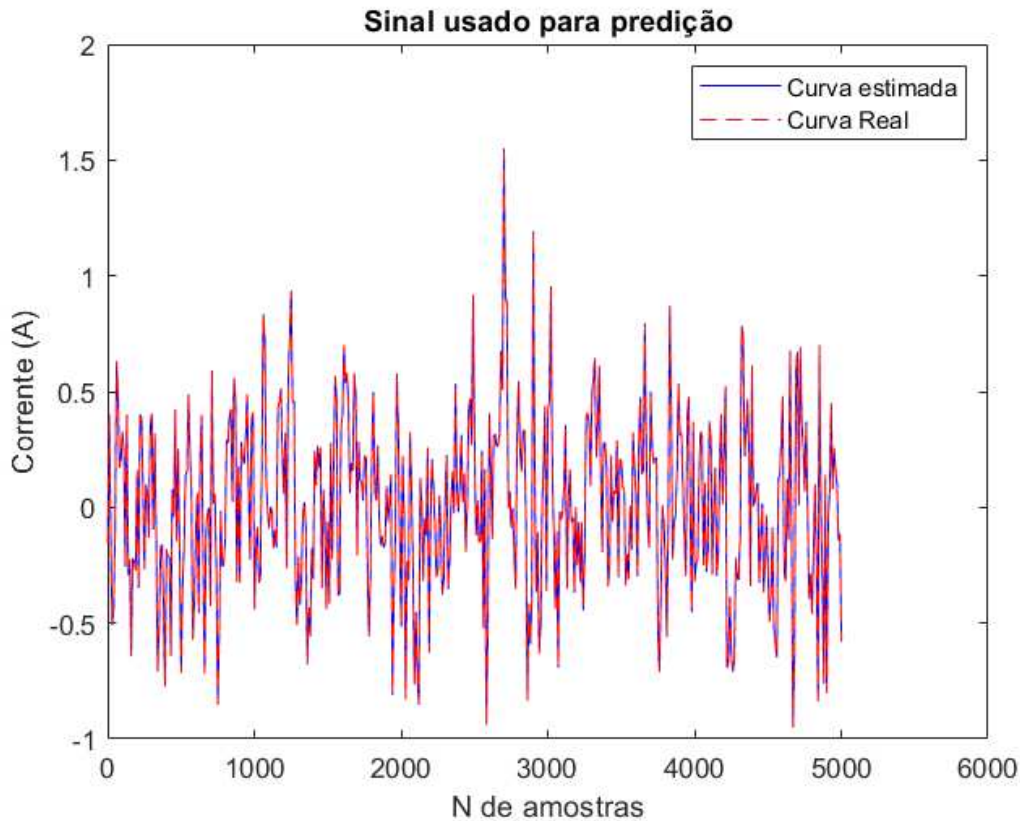
Figura 13 – Predição velocidade teste 03



Visualmente, tanto para a imagem 13 e 14, é possível observar poucas discordâncias entre as curvas. Só será possível ter noção da qualidade da predição com a medida do MSE. Para a



Figura 14 – Predição corrente teste 03



saída da velocidade o MSE é de  $9.31162 \times 10^{-6}$  e para a corrente o MSE foi de  $5.60831 \times 10^{-6}$ .

#### Teste 04

Em comparação ao teste 03 o número de neurônios na camada oculta foi reduzido, permanecendo a função de ativação a mesma.

- Número de neurônios na camada oculta = 25;
- Função de ativação = sigmoide;

Da mesma forma que no teste 03, conforme ilustrado nas figuras 15 e 16, só é possível analisar a qualidade da predição com os valores do MSE. Para a saída de velocidade, o MSE foi de  $4.49567 \times 10^{-5}$  e para a corrente o erro foi de  $4.16755 \times 10^{-5}$ . Utilizando o MSE como medida de comparação, os resultados da rede ELM batch foram melhores que os da rede MLP. Além disso, uma grande diminuição no número de neurônios da rede ELM, não resultou em uma considerável redução de performance.

Também foram realizados testes alterando a função de ativação, no caso a tangente hiperbólica, mas não houve nenhuma alteração minimamente significativa para ser relatada nas simulações. Na tabela 7 estão os resultados dos erros quadráticos médios para a variável de

Figura 15 – Predição velocidade teste 04

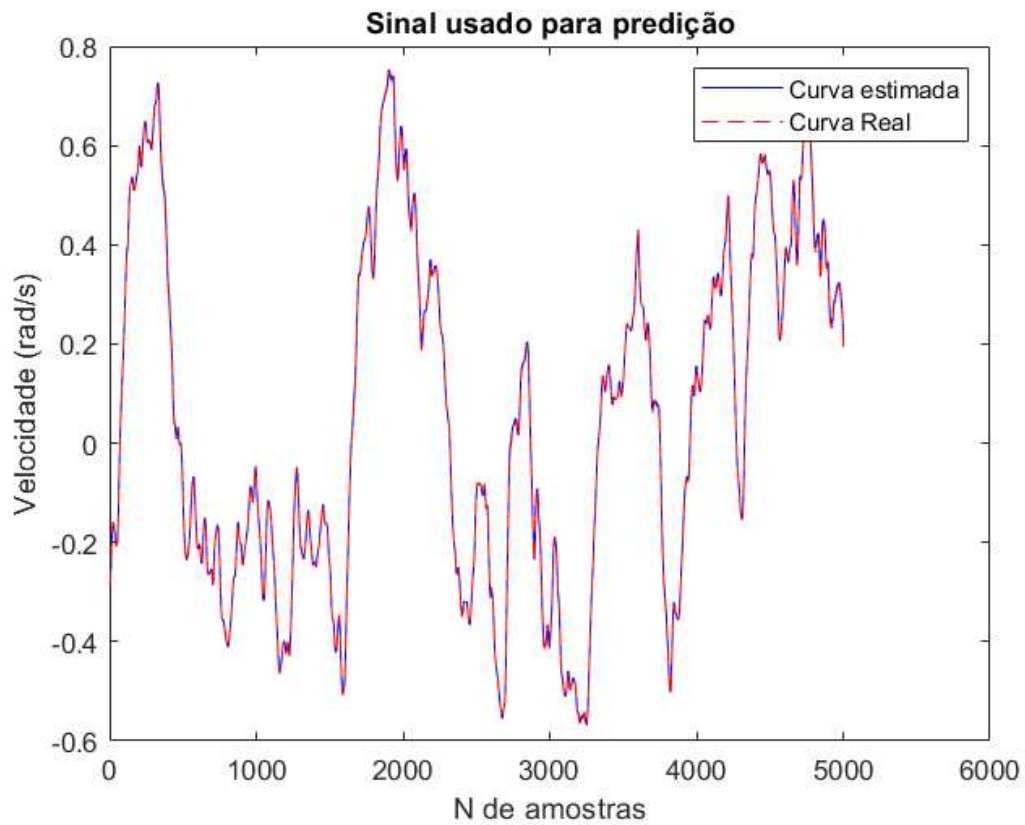
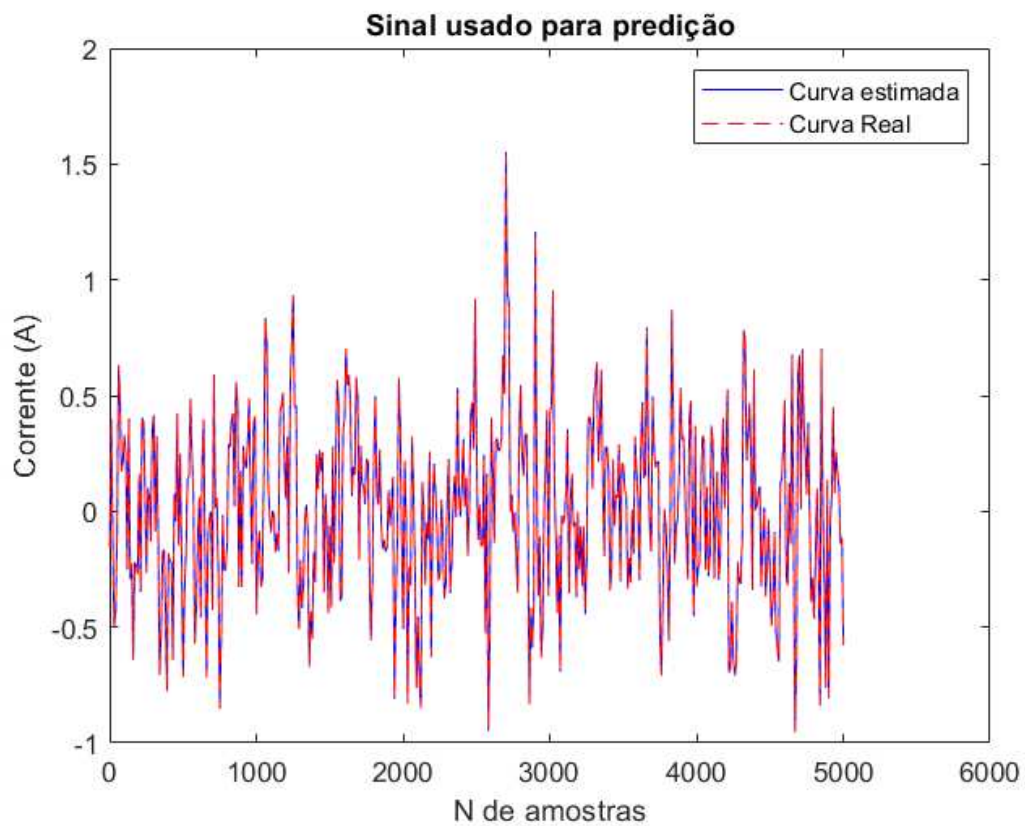


Figura 16 – Predição corrente teste 04



velocidade e corrente nos testes 03 e 04.

Tabela 7 – Resumo dos testes ELM Não recursivo entradas aleatórias

Variável de saída	Teste 03	Teste 04
Velocidade	$9.3116 \times 10^{-6}$	$4.4956 \times 10^{-5}$
Corrente	$5.6083 \times 10^{-6}$	$4.1675 \times 10^{-5}$

### 4.1.3 ELM Online

Nesta seção, são realizadas algumas simulações com os valores dos hiperparâmetros específicos da ELM Recursiva e cujo valores estão dentro do intervalo descrito na tabela 5.

#### Teste 05

- Número de neurônios na camada oculta = 25;
- Função de ativação = tangente hiperbólica;
- $P_{inicial} = 100000$ ;
- $W_{inicial} = 0.01$ .

A partir desses dados tem-se os resultados a seguir. A variável de saída da figura 17 é a velocidade e a variável de saída da imagem 18 é a corrente. Os gráficos representam a curva de saída real com a curva de valores preditos de acordo com a legenda presente na figura.

Figura 17 – Predição velocidade teste 05

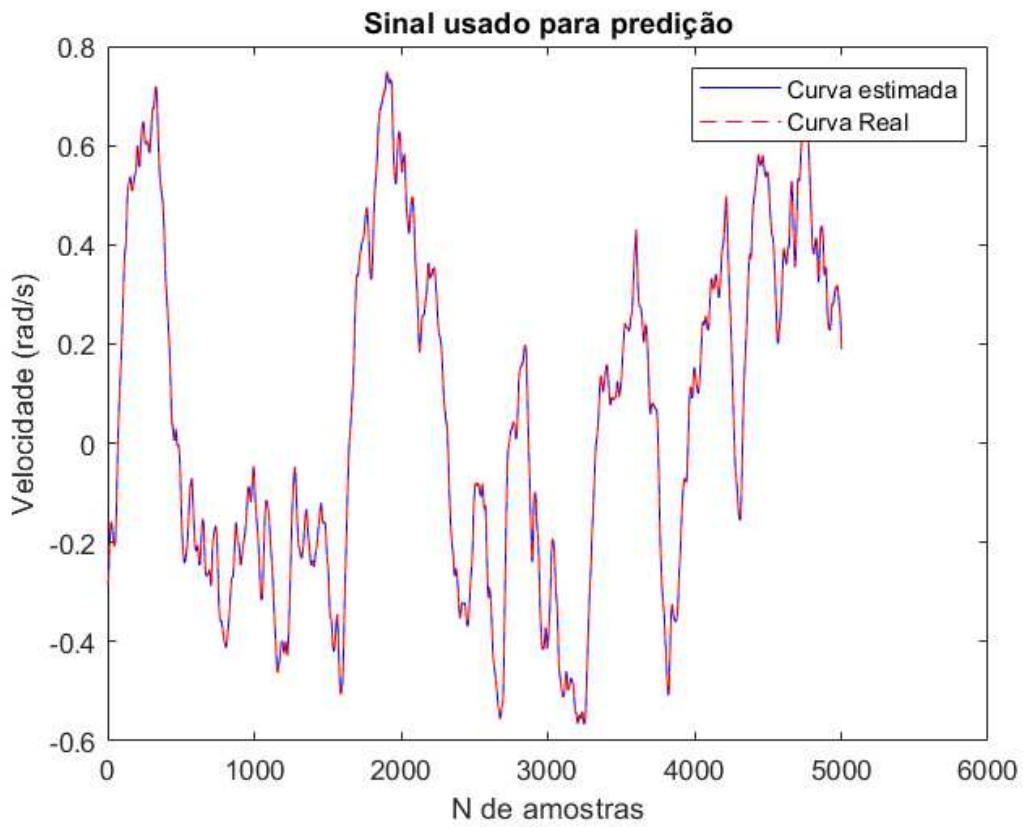
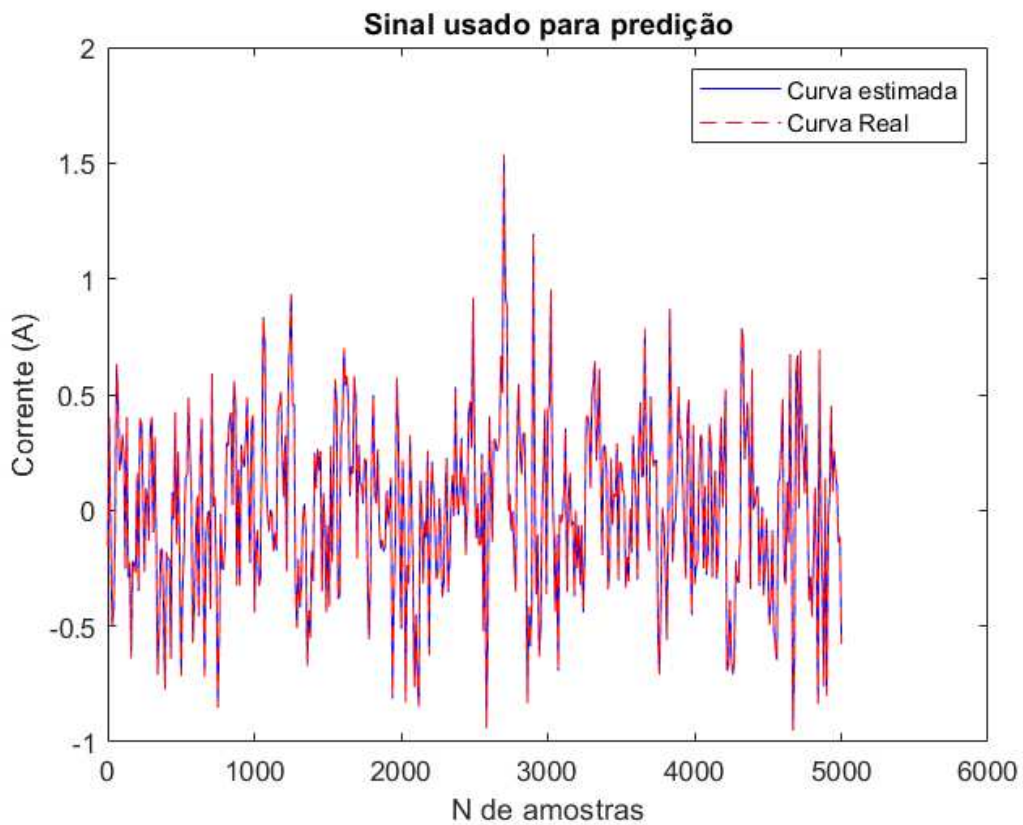


Figura 18 – Predição corrente teste 05



Só é possível distinguir algum erro com o valor de MSE. A qualidade da predição para as simulações com ELM Recursivo é superior comparado com os testes anteriores. Para a saída de velocidade na imagem 17 o MSE foi de  $2.995 \times 10^{-6}$  e para a corrente na imagem 18 o MSE foi de  $1.6433 \times 10^{-7}$ .

### Teste 06

Em comparação com o teste 05, foi alterado o hiperparâmetro  $P_{inicial}$ . Os outros valores e a função de ativação foram mantidos.

- Número de neurônios na camada oculta = 25;
- Função de ativação = tangente hiperbólica;
- $P_{inicial} = 1000$ ;
- $W_{inicial} = 0.01$ .

Nestes testes, o padrão de exibição dos gráficos de comparação das variáveis de saída foi mantido. Na figura 19 a variável é a velocidade e na figura 20 a variável é a corrente.

Figura 19 – Predição velocidade teste 06

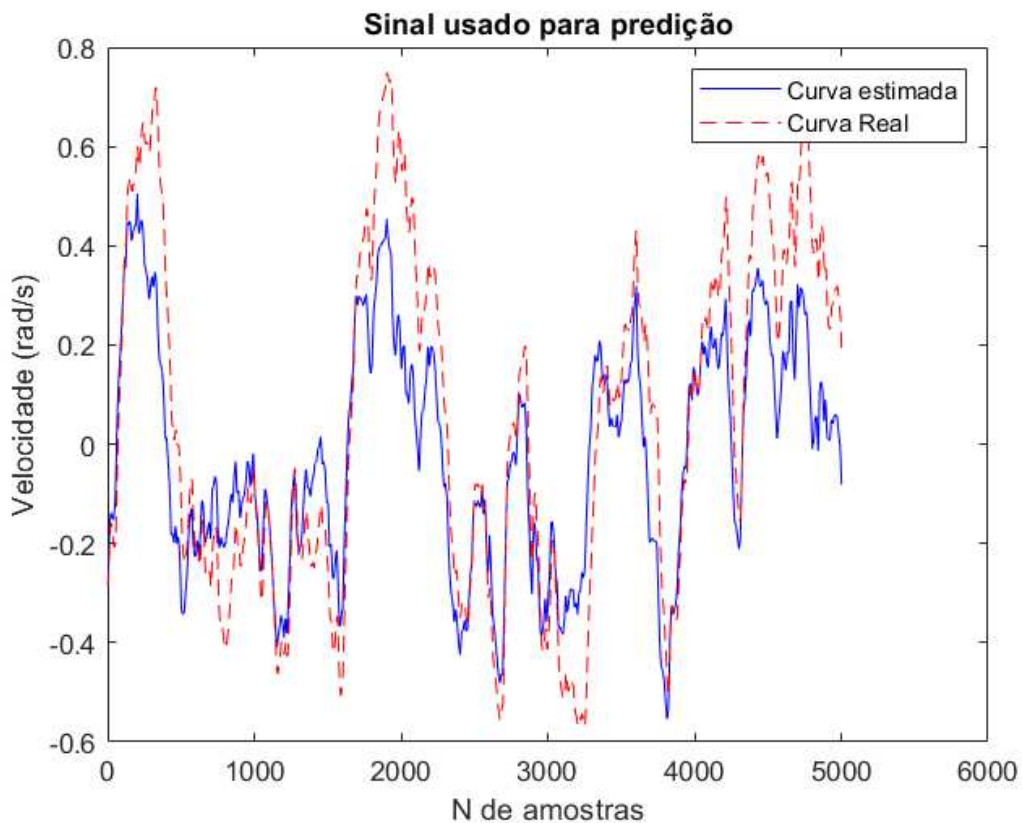
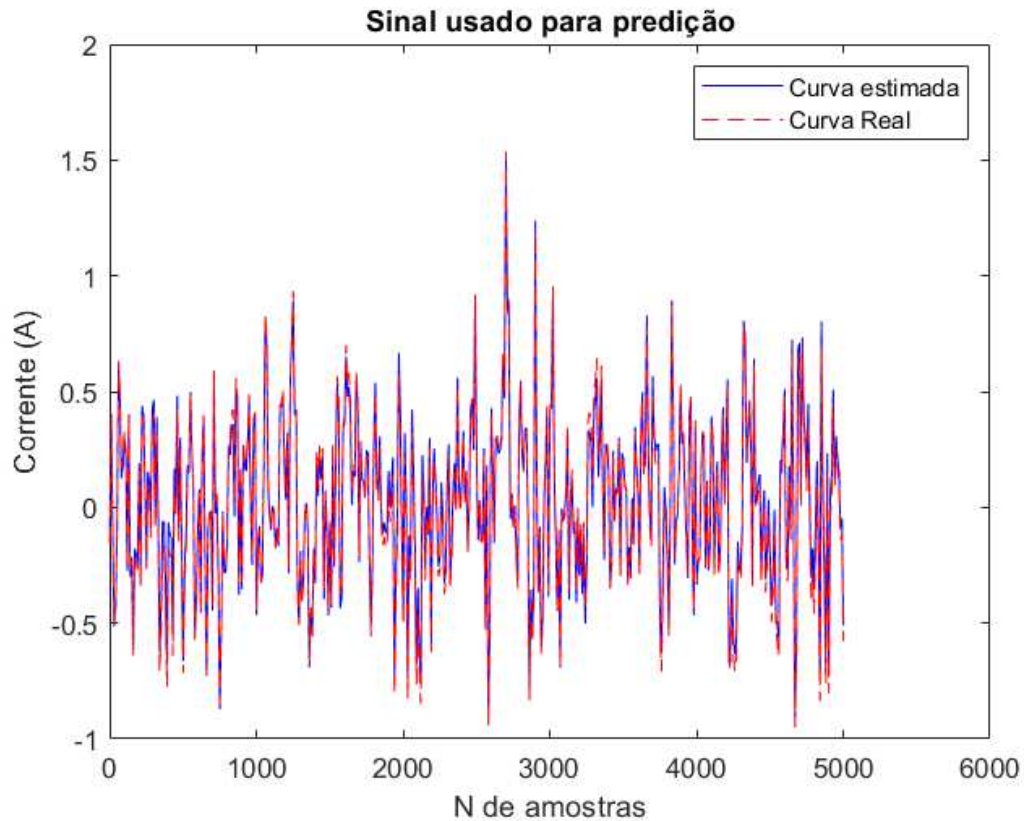


Figura 20 – Predição corrente teste 06



Visualmente, é possível observar grandes discordâncias entre as curvas dos valores preditos e os valores reais principalmente na figura 19 que representa a variável da velocidade. Já para a figura 20 que representa a corrente é possível observar alguma diferença entre as curvas, mas menos que na figura 19. Numericamente, o MSE da velocidade para essa simulação é de  $3.3377 \times 10^{-2}$ , e o MSE da corrente para essa simulação é de  $2.079 \times 10^{-3}$ .

#### Teste 07

Usando o teste 05 como base de comparação, o hiperparâmetro  $W_{inicial}$  foi alterado. Já os outros valores foram mantidos constantes em relação ao teste 5.

- Número de neurônios na camada oculta = 25;
- Função de ativação = tangente hiperbólica;
- $P_{inicial} = 100000$ ;
- $W_{inicial} = 0.001$ .



Figura 21 – Predição velocidade teste 07

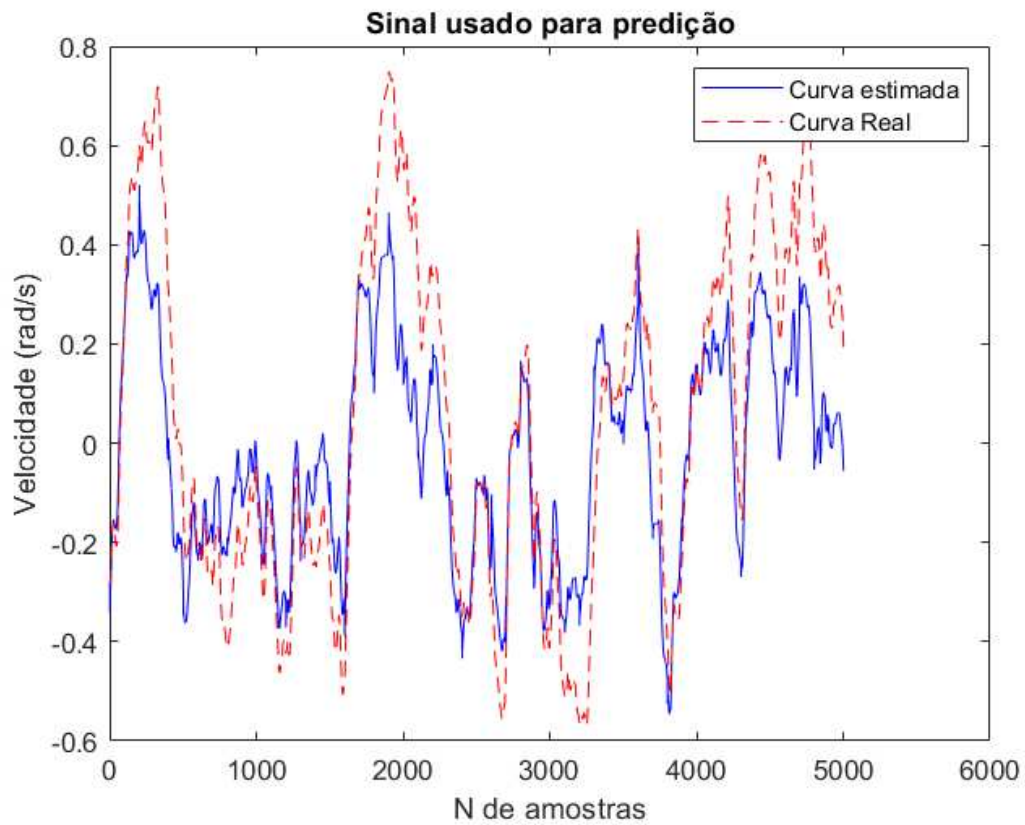
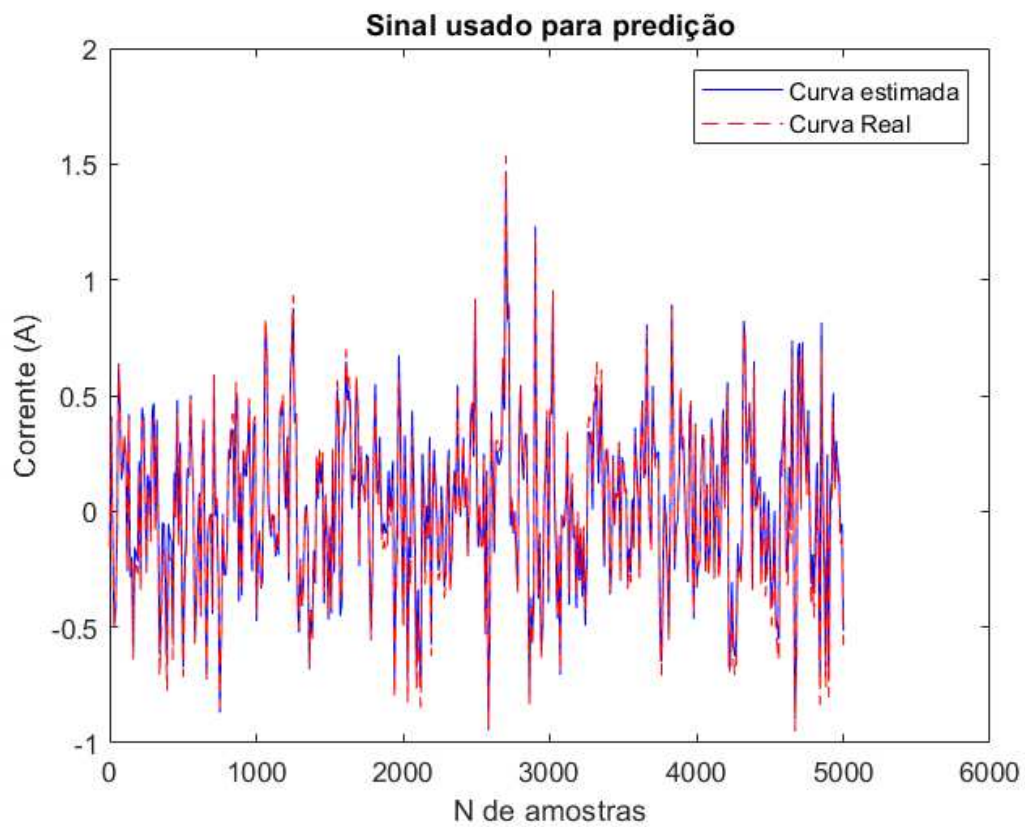


Figura 22 – Predição corrente teste 07



Assim como no teste 06, as observações são semelhantes. Na figura 21 a qual representa as curvas de velocidade, é possível observar diferenças significativas entre a curva dos valores reais e os valores preditos e com erro médio quadrático de  $2.6841 \times 10^{-2}$ . Já na figura 22 que representa as curvas da corrente é menos perceptível essa diferença, mas ainda possível observar alguns erros que são representados numericamente pelo erro de  $2.0676 \times 10^{-3}$ .

### Teste 08

Ainda usando o teste 05 como base, o hiperparâmetro modificado dessa vez foi a função de ativação, agora sendo utilizada a função sigmoide e mantendo os outros valores constantes.

- Número de neurônios na camada oculta = 25;
- Função de ativação = sigmoide;
- $P_{inicial} = 100000$ ;
- $W_{inicial} = 0.01$ .

Figura 23 – Predição velocidade teste 08

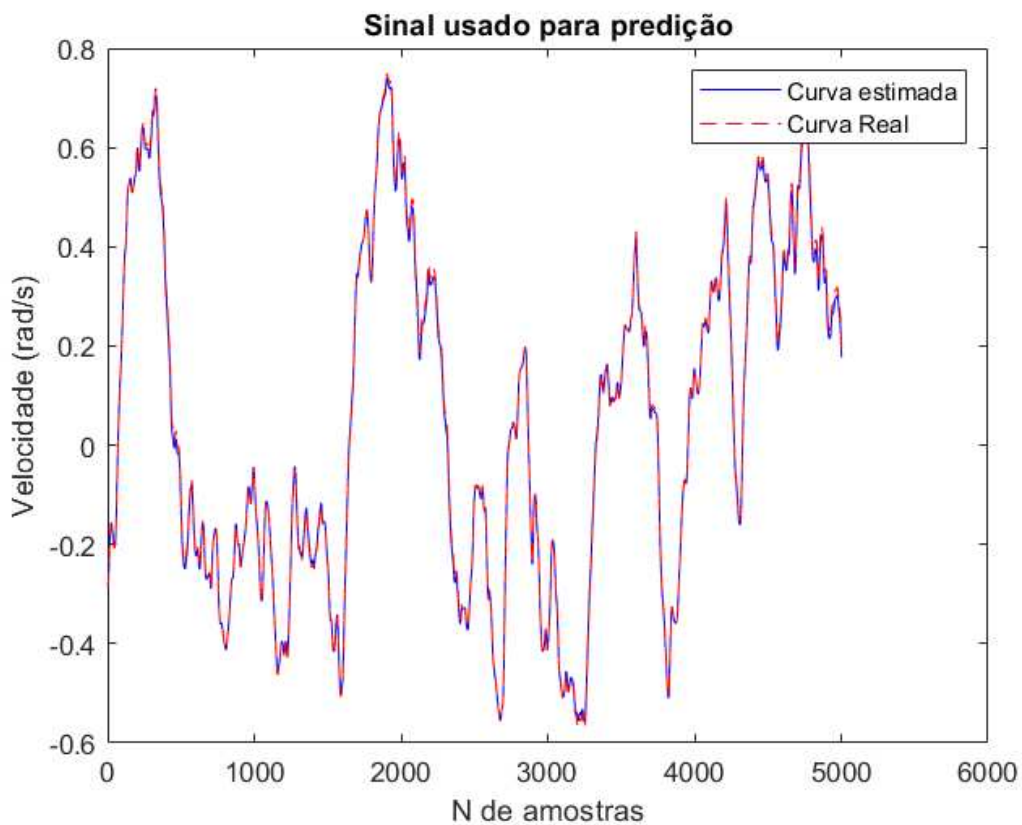
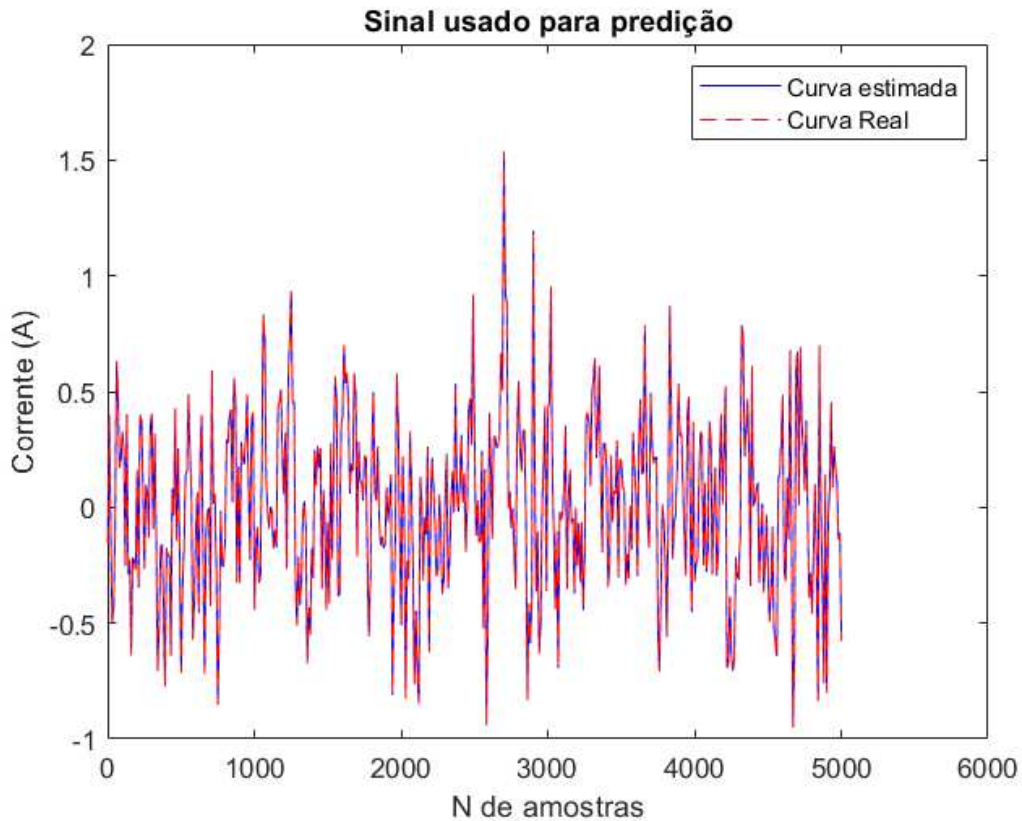




Figura 24 – Predição corrente teste 08



Nessa simulação um melhor resultado e poucas discrepâncias podem ser observadas evidenciando que, neste caso, as funções de ativação pouco influenciaram no resultado final. Com a métrica usada para quantificar isso, o MSE da figura 23 é de  $2.9156 \times 10^{-4}$  e o MSE da figura 24 é de  $2.0477 \times 10^{-5}$ .

#### Teste 09

Para finalizar esse ciclo de simulações, e ainda usando o teste 05 como base, agora foi alterado o número de neurônios na camada oculta, e foram mantidos os valores dos demais hiperparâmetros do teste 5.

- Número de neurônios na camada oculta = 50;
- Função de ativação = tangente hiperbólica;
- $P_{inicial} = 100000$ ;
- $W_{inicial} = 0.01$ .

Figura 25 – Predição velocidade teste 09

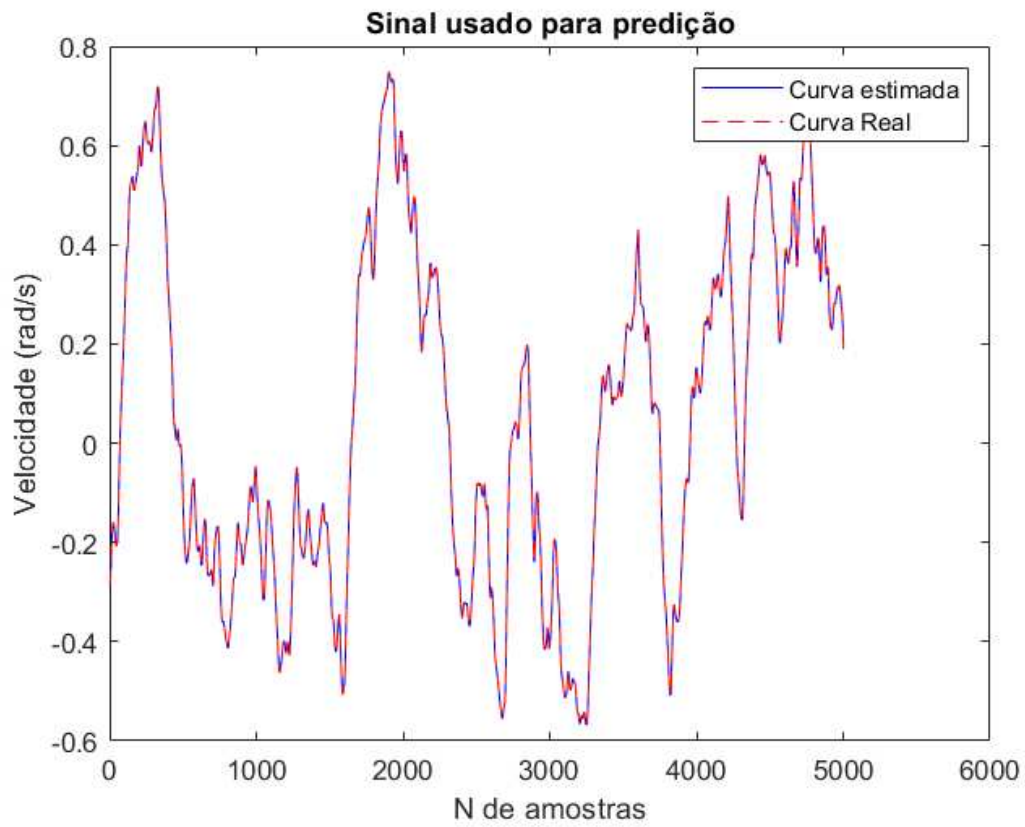
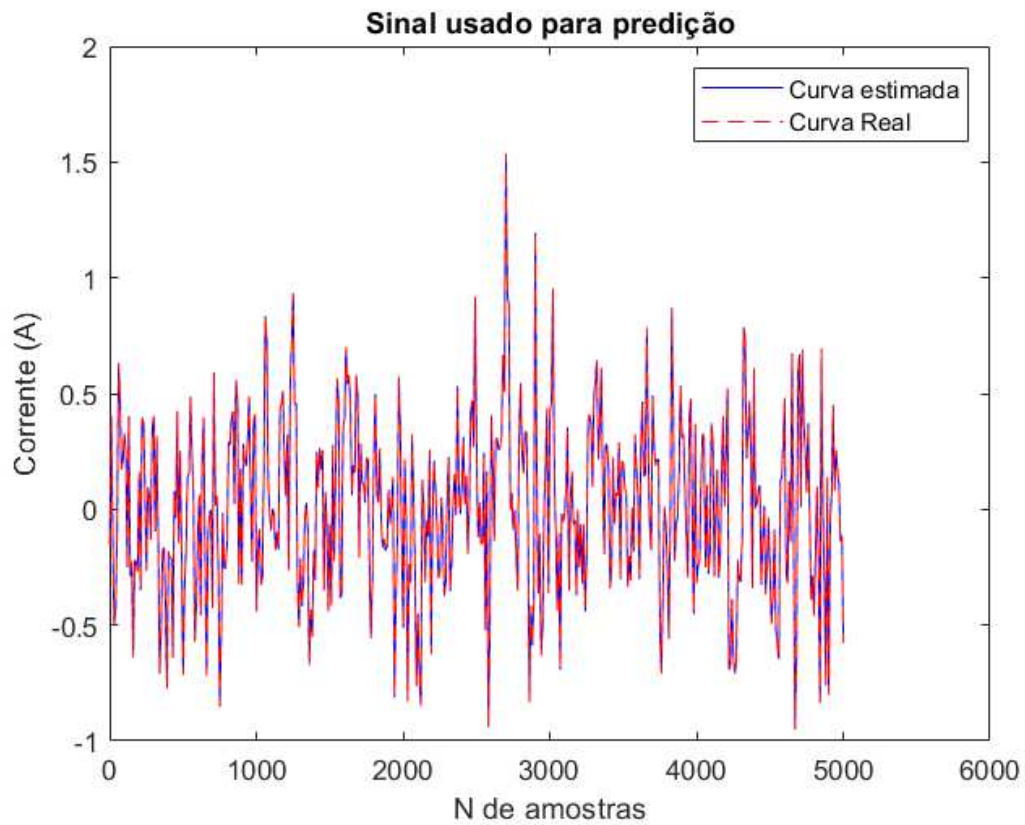


Figura 26 – Predição corrente teste 09



Na figura 25 o MSE foi de  $1.9969 \times 10^{-6}$  e na figura 26 o MSE foi de  $7.2347 \times 10^{-8}$ , representado uma qualidade muito boa da predição nesse teste com os hiperparâmetros selecionados.

A partir das observações dos testes 05 ao 09 e de acordo com os métodos algumas observações podem ser feitas. A primeira delas, assim como na MLP, é sobre a importância da definição dos hiperparâmetros para a rede  $ELM_o$ . Os resultados apresentam diferenças consideráveis na medida quantitativa de erro médio quadrático de acordo com os hiperparâmetros escolhidos. Um resumo dos resultados é mostrado na tabela 8.

Tabela 8 – Resumo dos teste ELM Recursivo com entradas aleatórias

Variável de saída	Teste 05	Teste 06	Teste 07	Teste 08	Teste 09
Velocidade	$2.995 \times 10^{-6}$	$3.337 \times 10^{-2}$	$2.684 \times 10^{-2}$	$2.915 \times 10^{-4}$	$1.996 \times 10^{-6}$
Corrente	$1.643 \times 10^{-7}$	$2.079 \times 10^{-3}$	$2.067 \times 10^{-3}$	$2.047 \times 10^{-5}$	$7.234 \times 10^{-8}$

Dentre as simulações descritas, percebe-se a sensibilidade dos hiperparâmetros  $W_{inicial}$  e  $P_{inicial}$  que foram explicados na seção 3.4.3. A  $ELM_o$  depende diretamente desses valores que são escolhidos por tentativa e erro. Dentro do intervalo de valores para  $W_{inicial}$  e  $P_{inicial}$ , quanto maior os valores dentro do intervalo melhor a aproximação dos valores simulados com o banco de dados inicial. Para evidenciar a importância dos outros hiperparâmetros presentes na  $ELM_o$ , no teste 07 foi alterado a função de ativação em relação ao teste 05 e não houve mudança significativa na grandeza do erro médio quadrático. E no teste 08 foi alterado o número de neurônios na camada oculta em relação ao teste 05 e houve uma diferença pouco significativa nos resultados. Para a ELM Recursiva, a importância dos pesos e do bias como descritos na seção 2.3.4.2 foram observados nos testes realizados. A escolha devida desses valores iniciais afetará diretamente na qualidade do treinamento e da consequente predição.

Para o banco de dados onde as variáveis de entrada, tensão e carga são sinais aleatórios e usando as redes neurais apresentadas nesse trabalho, podemos resumir todas as simulações realizadas na tabela abaixo:

A rede neural ELM apresenta, com as devidas otimizações dos hiperparâmetros relacionados, os melhores resultados. E dentre a variação da recursividade ou não da ELM, os melhores resultados são do algoritmo recursivo, porém este possui mais hiperparâmetros a serem ajustados. Dentre os teste realizados com o banco de dados com entradas de sinal aleatório, o Teste 09 teve o menor erro, cujo resultado esta destacado em negrito na tabela 9

Tabela 9 – Resumo dos teste com entradas aleatórias

		ERRO QUADRÁTICO MÉDIO	
		VELOCIDADE	CORRENTE
MLP	TESTE 01	$1.7390 \times 10^{-2}$	$5.8418 \times 10^{-3}$
	TESTE 02	$1.7463 \times 10^{-1}$	$1.1483 \times 10^{-2}$
ELM BATCH	TESTE 03	$9.3116 \times 10^{-6}$	$5.6083 \times 10^{-6}$
	TESTE 04	$4.4956 \times 10^{-5}$	$4.1675 \times 10^{-5}$
ELM ONLINE	TESTE 05	$2.9950 \times 10^{-6}$	$1.6433 \times 10^{-7}$
	TESTE 06	$3.3377 \times 10^{-2}$	$2.0795 \times 10^{-3}$
	TESTE 07	$2.684 \times 10^{-2}$	$2.067 \times 10^{-3}$
	TESTE 08	$2.9156 \times 10^{-4}$	$2.0477 \times 10^{-5}$
	<b>TESTE 09</b>	$2.5733 \times 10^{-6}$	<b><math>1.0255 \times 10^{-7}</math></b>

## 4.2 Sinal de entrada de tensão e carga APRBS

### *MLP*

Como descrito no capítulo de materiais e métodos, os hiperparâmetros da MLP variam em determinados valores de acordo com a tabela 3. Para esse banco de dados de entrada foram realizados apenas dois testes como descritos a seguir.

### *Teste 10*

Para o teste 10 foram utilizados os seguintes hiperparâmetros:

- Número de épocas = 60;
- Número de neurônios na camada oculta = 10;
- Função de ativação = sigmoide;
- Taxa de aprendizagem = 0.05.

A partir desses hiperparâmetros temos os seguintes resultados. As figuras 27 e 28 apresenta curvas sobrepostas dos valores reais do banco de dados e dos dados preditos com a metodologia usada e os hiperparâmetros selecionados. A variável da figura 27 é a velocidade e a variável de saída da figura 28 é a corrente.

Figura 27 – Predição velocidade teste 10

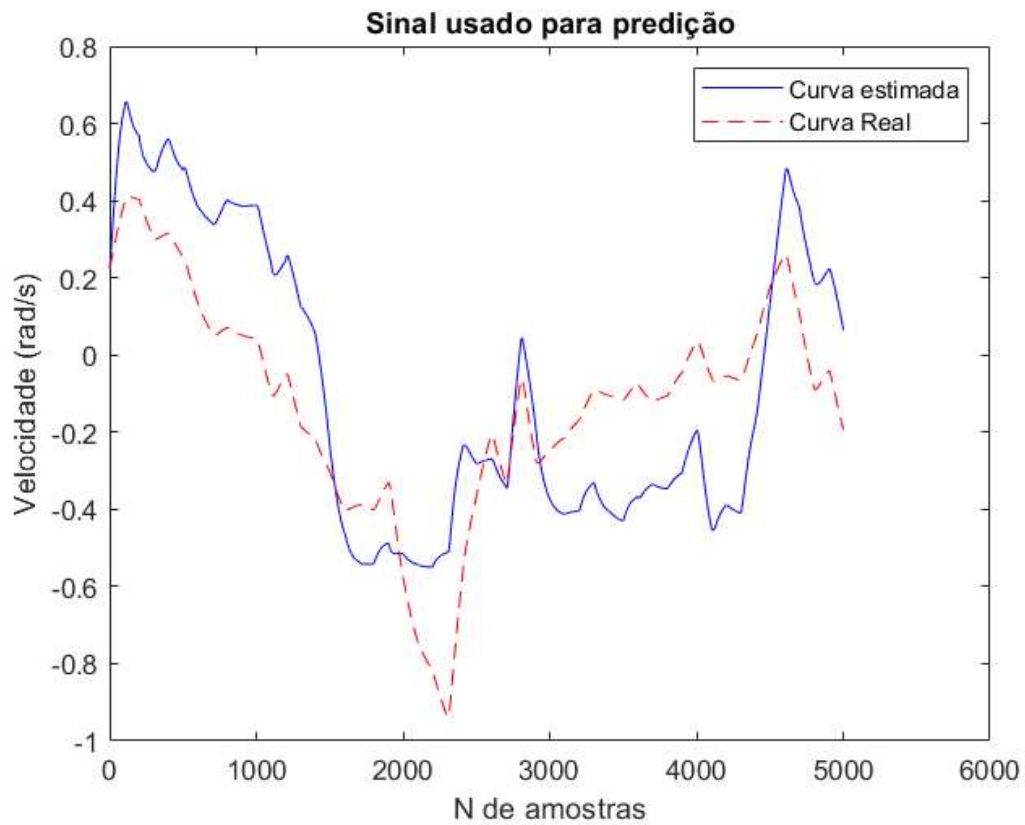
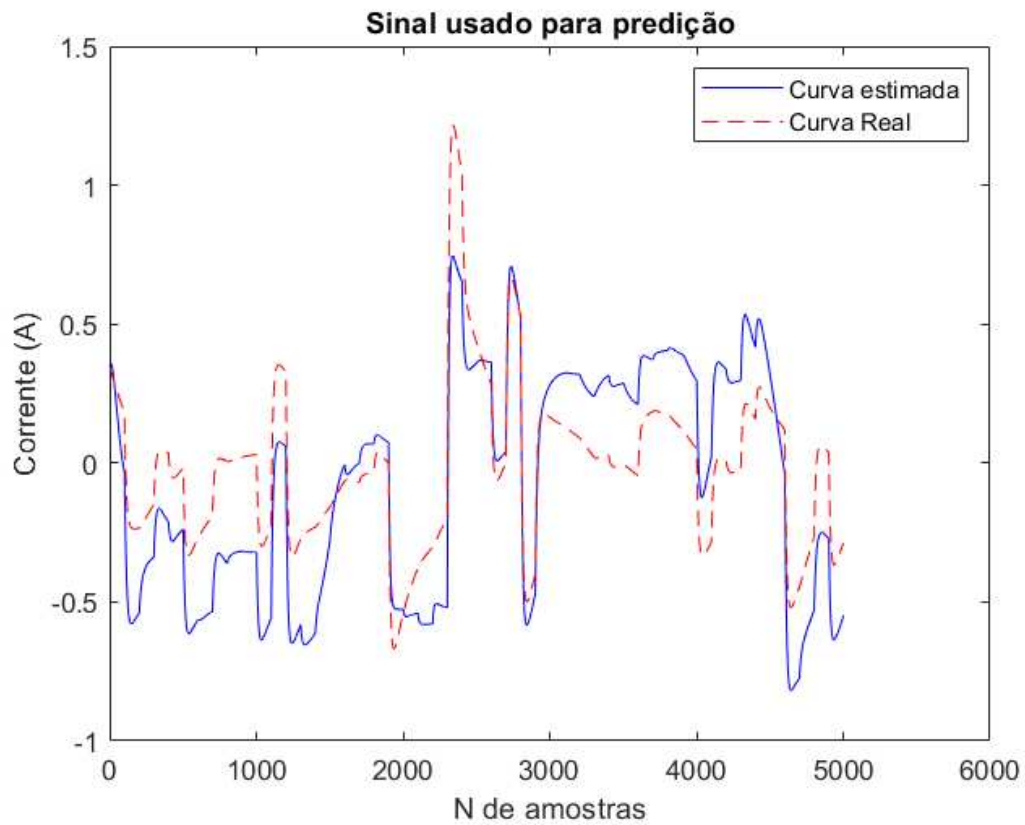


Figura 28 – Predição corrente teste 10



Visualmente, há uma discordância considerável entre as curvas nos valores preditos da velocidade. Nos valores preditos de corrente em relação aos valores reais há uma diferença visível, mas menor em relação a figura 27. Numericamente, com a medida para quantificar essa discordância, o MSE da velocidade para o teste 10 é de  $6.5300 \times 10^{-2}$  e o MSE para a corrente é de  $5.9904 \times 10^{-2}$ .

### Teste 11

Em comparação ao teste 10, essa simulação apresenta apenas uma mudança no número de neurônios na camada oculta, mantendo os outros hiperparâmetros inalterados.

- Número de épocas = 60;
- Número de neurônios na camada oculta = 20;
- Função de ativação = sigmoide;
- Taxa de aprendizagem = 0.05.

A variável da figura 29 é a velocidade e a variável de saída da figura 30 é a corrente.

Figura 29 – Prerdição velocidade teste 11

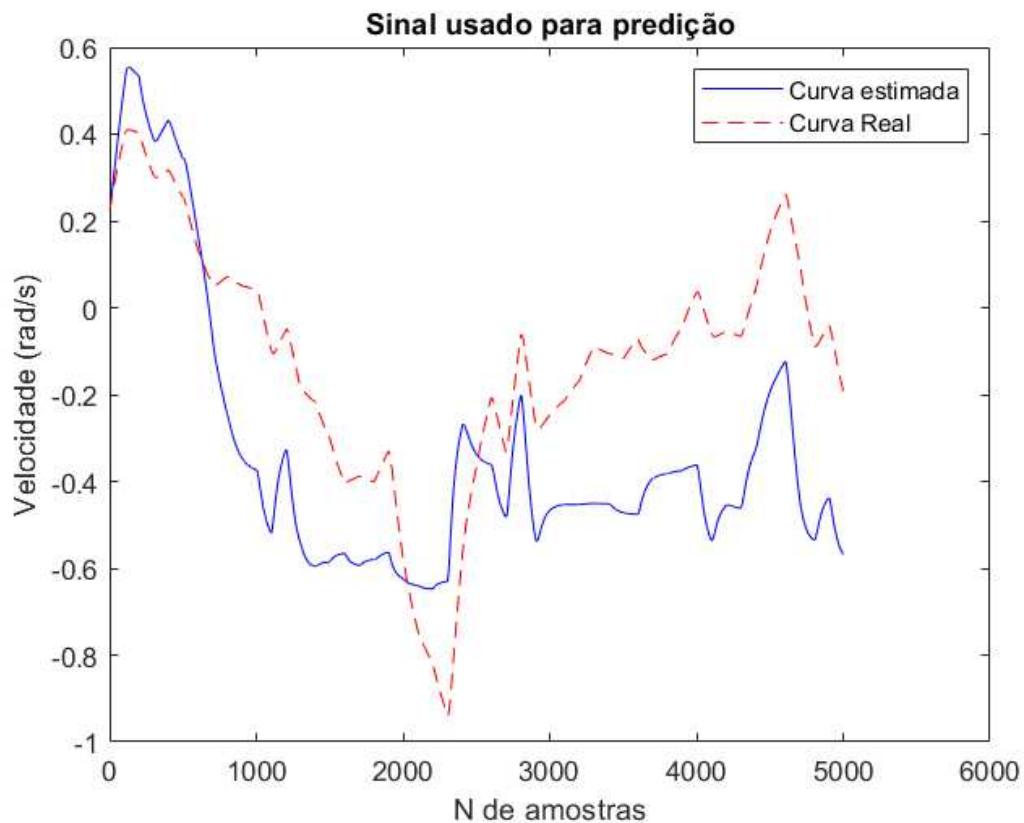
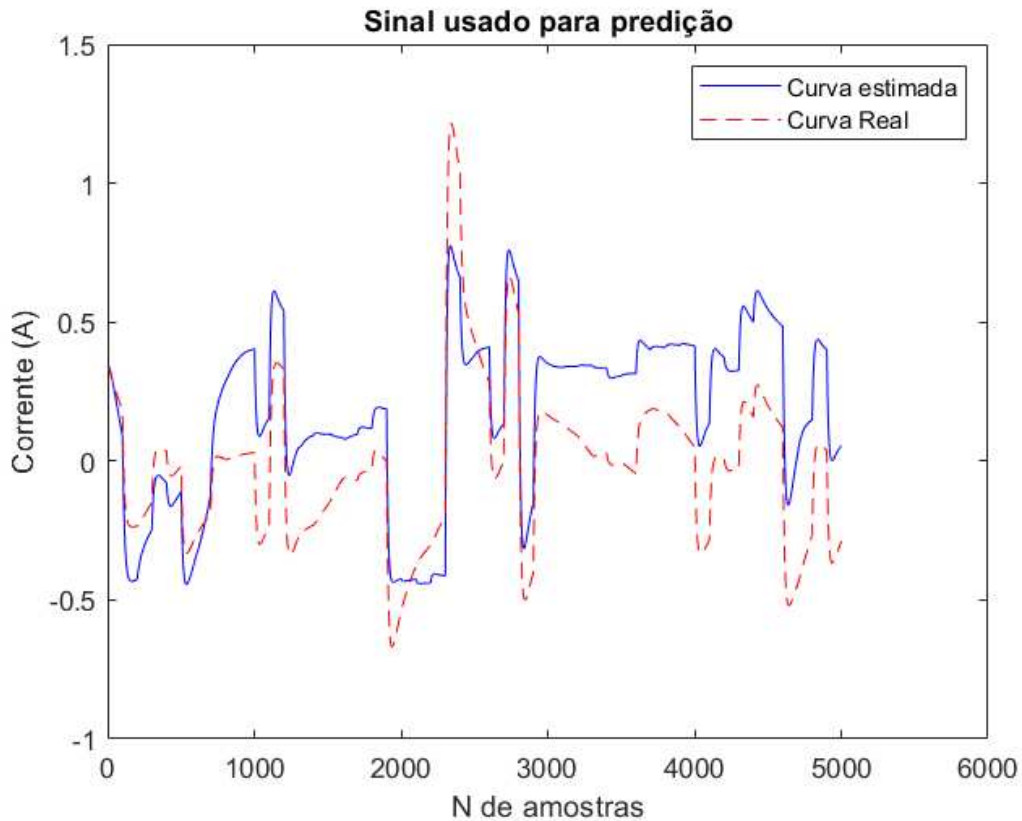


Figura 30 – Predição corrente teste 11



Nesses resultados também se percebe grandes discordâncias na saída de velocidade quando analisadas visualmente na figura 29. Já para a corrente na figura 30, as diferenças ocorrem em alguns picos e podem ser observadas. Numericamente, o erro quadrático da velocidade é de  $1.1670 \times 10^{-1}$  e da corrente é de  $\times 10^{-2}$ .

A partir dos testes 10 e 11, onde foram usados os mesmos dados de entrada e saída, pode-se inferir que, um aumento no número de neurônios da camada oculta não resultou em uma melhora significativa na ordem de grandeza do erro quadrático médio, como pode-se observar na Tabela 10

Tabela 10 – Resumo dos teste MLP com entradas APRBS

Variável de saída	Teste 10	Teste 11
Velocidade	$6.5300 \times 10^{-2}$	$1.1670 \times 10^{-1}$
Corrente	$5.9904 \times 10^{-2}$	$8.9782 \times 10^{-2}$

### 4.2.1 ELM Não recursivo

Usaremos agora a rede neural  $ELM_b$ , e para essa rede os hiperparâmetros são os descritos na tabela4

#### Teste 12

- Número de neurônios na camada oculta = 50;
- Função de ativação = sigmoide;

A partir desses dados tem-se os resultados a seguir, a variável de saída da figura 31 é a velocidade e a variável de saída da 32 é a corrente. Os gráficos representam a curva de saída real com a curva de valores preditos de acordo com a legenda presente na imagem.

Figura 31 – Predição velocidade teste 12

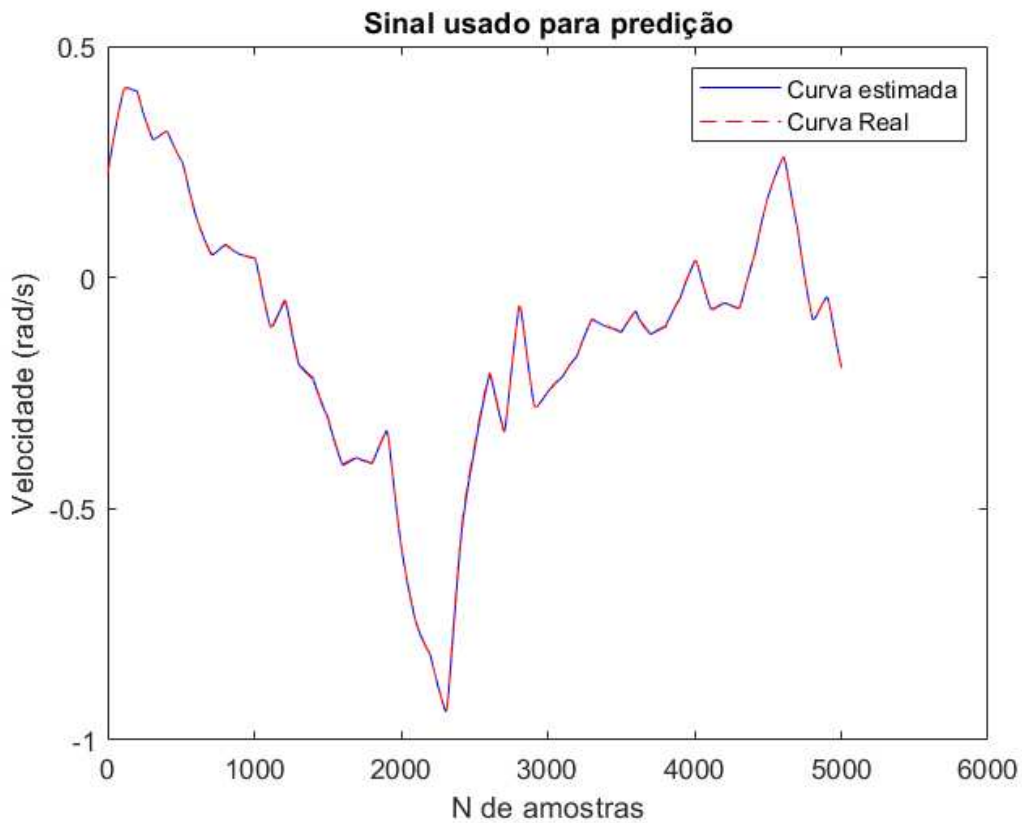
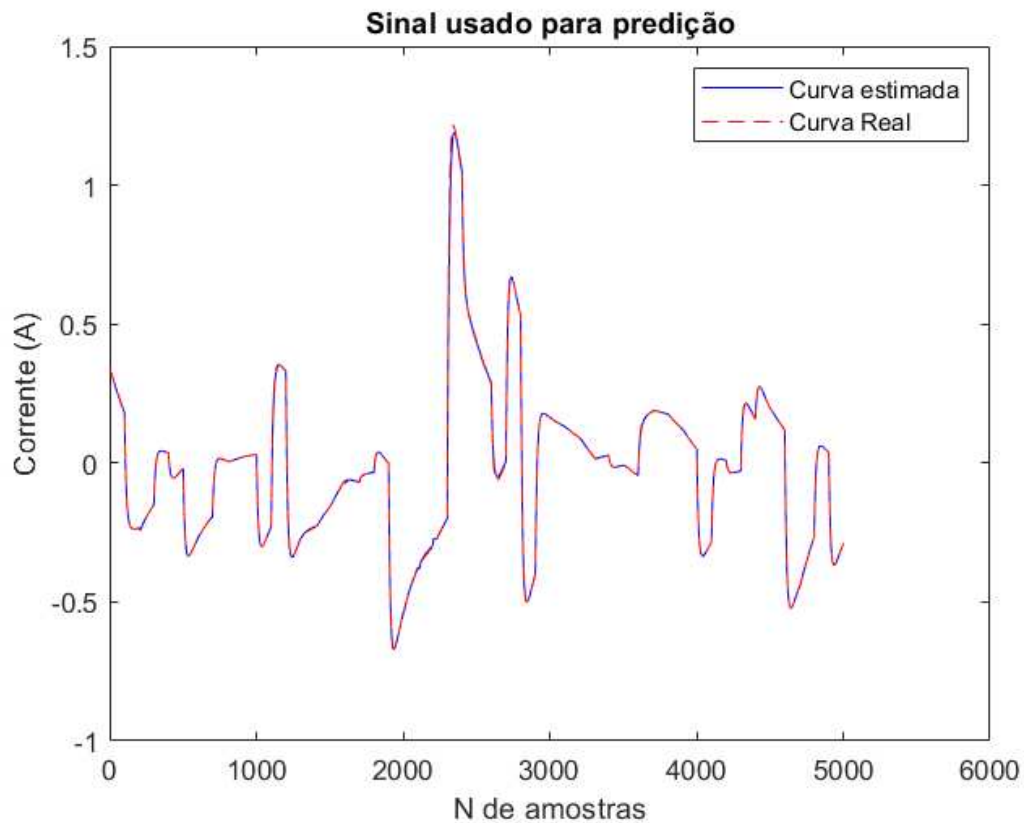




Figura 32 – Predição corrente teste 12



Visualmente, tanto para as figuras 31 e 32, é possível observar poucas discordâncias entre as curvas. Só será possível ter noção da qualidade da predição com a medida do MSE. Para a saída da velocidade o MSE é de  $\times 10^{-6}$  e para a corrente o MSE foi de  $3.1540 \times 10^{-5}$ .

### Teste 13

Em comparação ao teste 12 o número de neurônios na camada oculta foi alterado, permanecendo a função de ativação a mesma.

- Número de neurônios na camada oculta = 25;
- Função de ativação = sigmoide;

Figura 33 – Predição velocidade teste 13

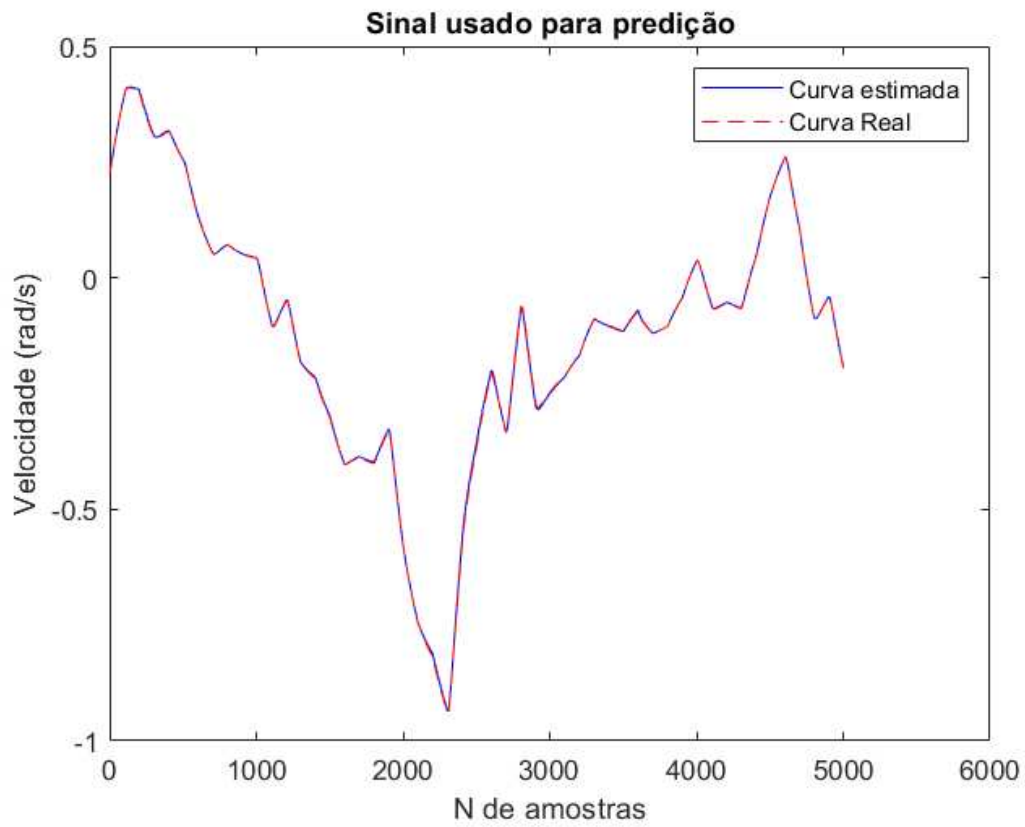
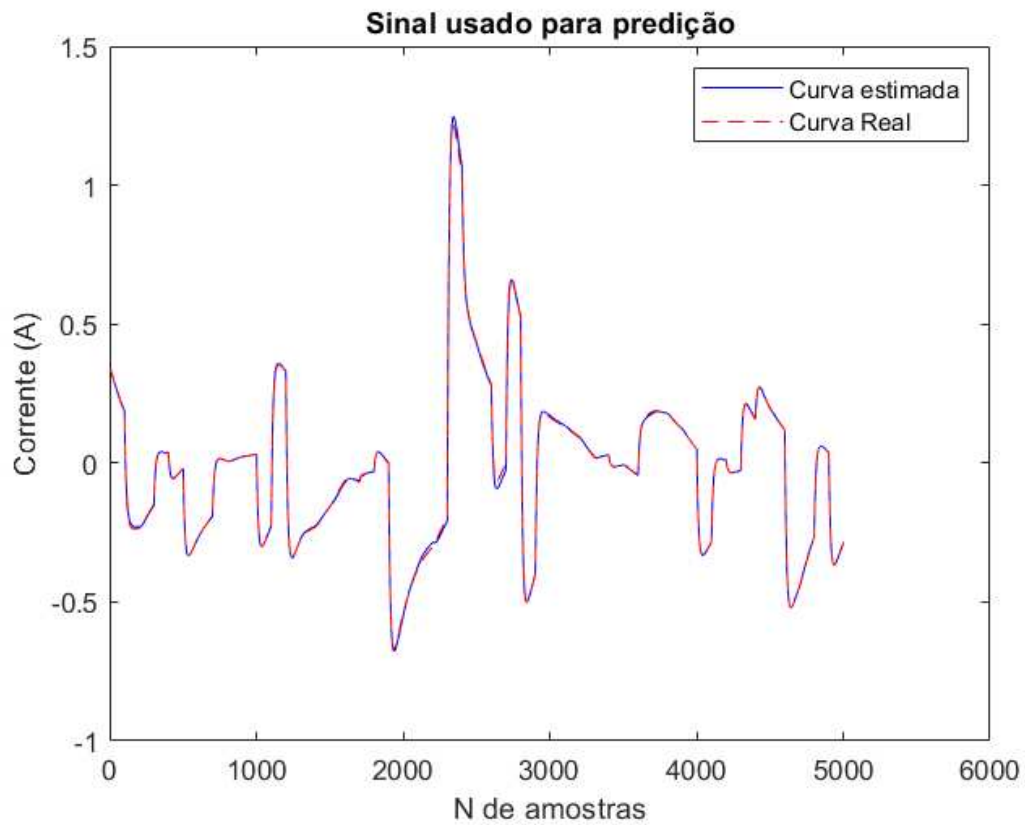


Figura 34 – Predição corrente teste 13



Assim como no teste 12, só é possível analisar a qualidade da predição com os valores do MSE. Para a saída de velocidade o erro foi de  $4.6309 \times 10^{-6}$  e para a corrente o erro foi de  $1.3260 \times 10^{-5}$ . Utilizando a medida MSE, os resultados da rede ELM Batch foram melhores que os da rede MLP. Além disso, esta rede possui menos hiperparâmetros que a MLP. Por fim, mesmo com uma redução considerável de neurônios na camada oculta, os resultados com a rede ELM mantiveram baixos níveis de MSE. Foram realizados testes com uma função de ativação diferente, no caso a tangente hiperbólica, mas não houve nenhuma alteração minimamente significativa para ser relatada nas simulações. A tabela 11 mostra os resultados dos erros quadráticos médios para a variável de velocidade e corrente nos testes 12 e 13.

Tabela 11 – Resumo dos teste ELM Não recursivo com entradas APRBS

Variável de saída	Teste 12	Teste 13
Velocidade	$2.2339 \times 10^{-4}$	$4.6309 \times 10^{-6}$
Corrente	$3.1540 \times 10^{-5}$	$1.3260 \times 10^{-5}$

#### 4.2.2 ELM Online

Como descrito na seção 3.4.3, serão realizados algumas simulações com os valores dos hiperparâmetros específicos da ELM Recursiva e com os valores dentro do intervalo descrito na tabela 5.

##### Teste 14

- Número de neurônios na camada oculta = 25;
- Função de ativação = tangente hiperbólica;
- $P_{inicial} = 100000$ ;
- $W_{inicial} = 0.01$

A partir desses dados tem-se os resultados a seguir, a variável de saída da figura 35 é a velocidade e a variável de saída da imagem 36 é a corrente. Os gráficos representam a comparação entre a curva de saída real com a curva de valores preditos de acordo com a legenda presente na imagem.

Figura 35 – Predição velocidade teste 14

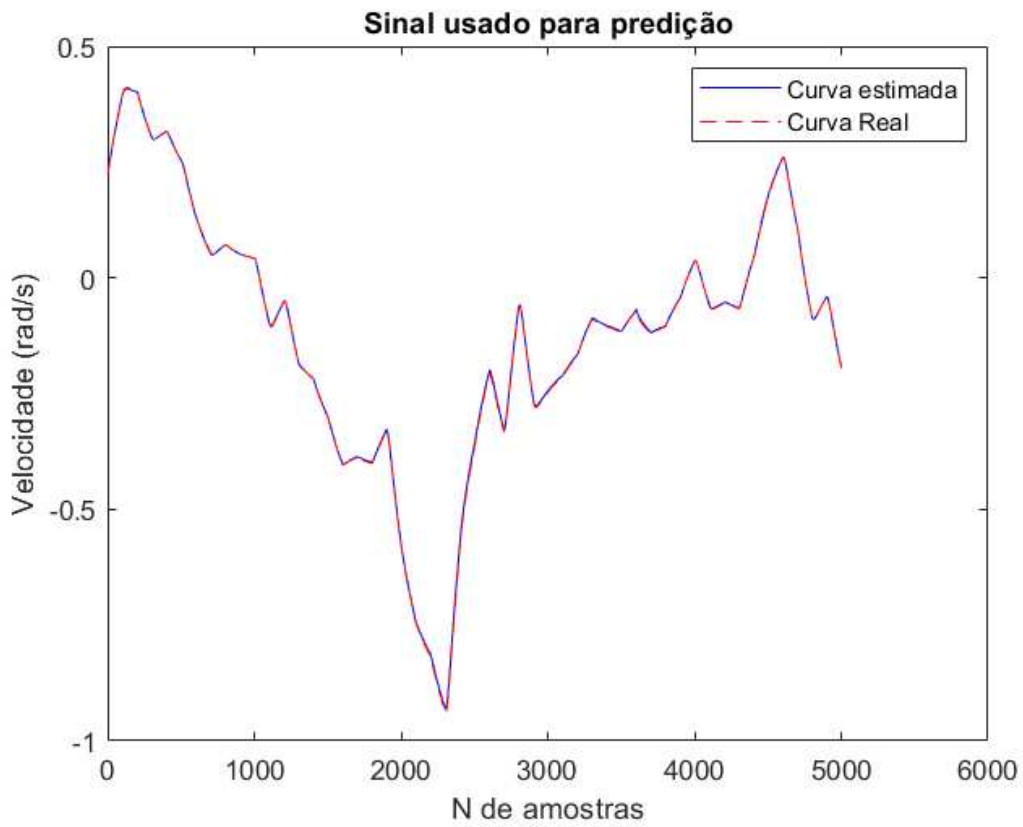
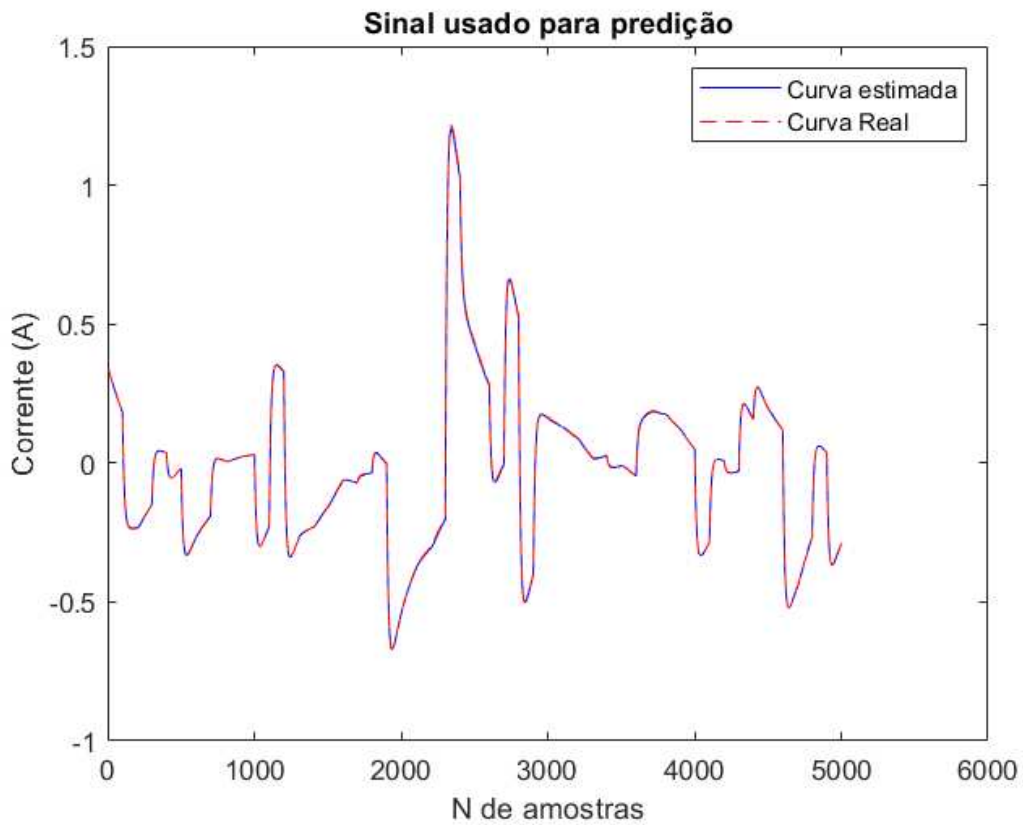


Figura 36 – Predição corrente teste 14



Só é possível distinguir algum erro com o valor de MSE. A qualidade da predição para as simulações com ELM Recursivo é superior comparado com os testes anteriores. Para a saída de velocidade na imagem 35 o MSE foi de  $1.4804e \times 10^{-5}$  e para a corrente na imagem 36 o MSE foi de  $1.3577 \times 10^{-5}$ .

### Teste 15

Em comparação com o teste 14, foi alterado o hiperparâmetro  $P_{inicial}$ . Os outros valores e a função de ativação foram mantidos.

- Número de neurônios na camada oculta = 25;
- Função de ativação = tangente hiperbólica;
- $P_{inicial} = 1000$ ;
- $W_{inicial} = 0.01$

O padrão de exibição dos gráficos de comparação das variáveis de saída, para estes testes, foram mantidos. Na imagem 37 a variável é a velocidade e na imagem 38 a variável é a corrente.

Figura 37 – Predição velocidade teste 15

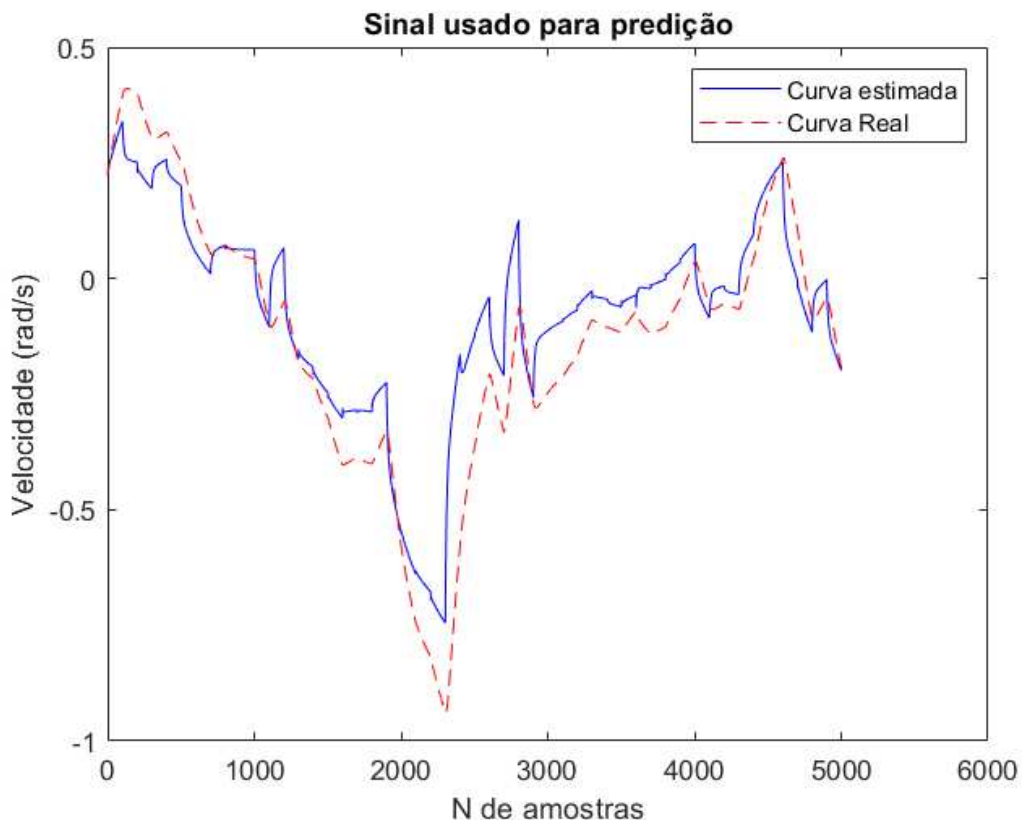
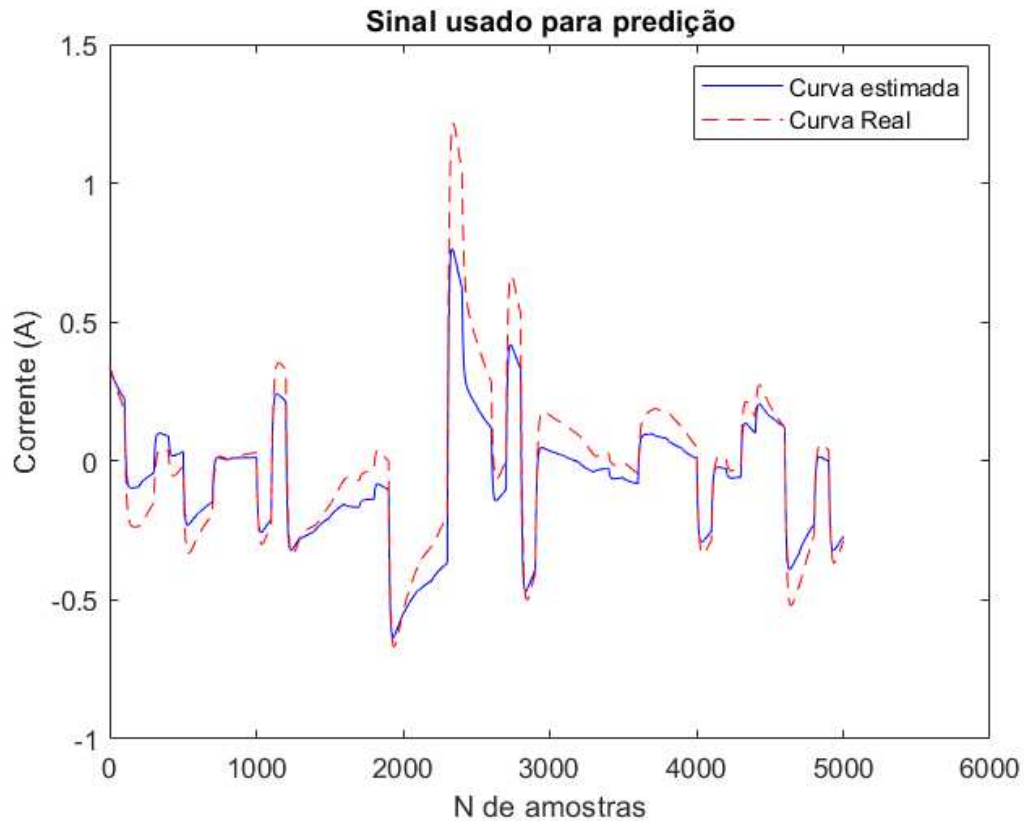


Figura 38 – Predição corrente teste 15



Visualmente, é possível observar grandes discordâncias entre as curvas dos valores preditos e os valores reais principalmente na imagem 37 que representa a variável da velocidade. Já para a imagem 38 que representa a corrente é possível observar alguma diferença entre as curvas, mas menos que que na imagem 37. Numericamente, o MSE da velocidade para essa simulação é de  $3.99071 \times 10^{-2}$ , e o MSE da corrente para essa simulação é de  $3.3916 \times 10^{-2}$ .

#### Teste 16

Usando o teste 14 como base de comparação, agora foi alterado o hiperparâmetro o  $W_{inicial}$ . Os outros valores e função de ativação foram mantidos os mesmo em relação ao teste 14.

- Número de neurônios na camada oculta = 25;
- Função de ativação = tangente hiperbólica;
- $P_{inicial} = 100000$ ;
- $W_{inicial} = 0.001$

Figura 39 – Predição velocidade teste 16

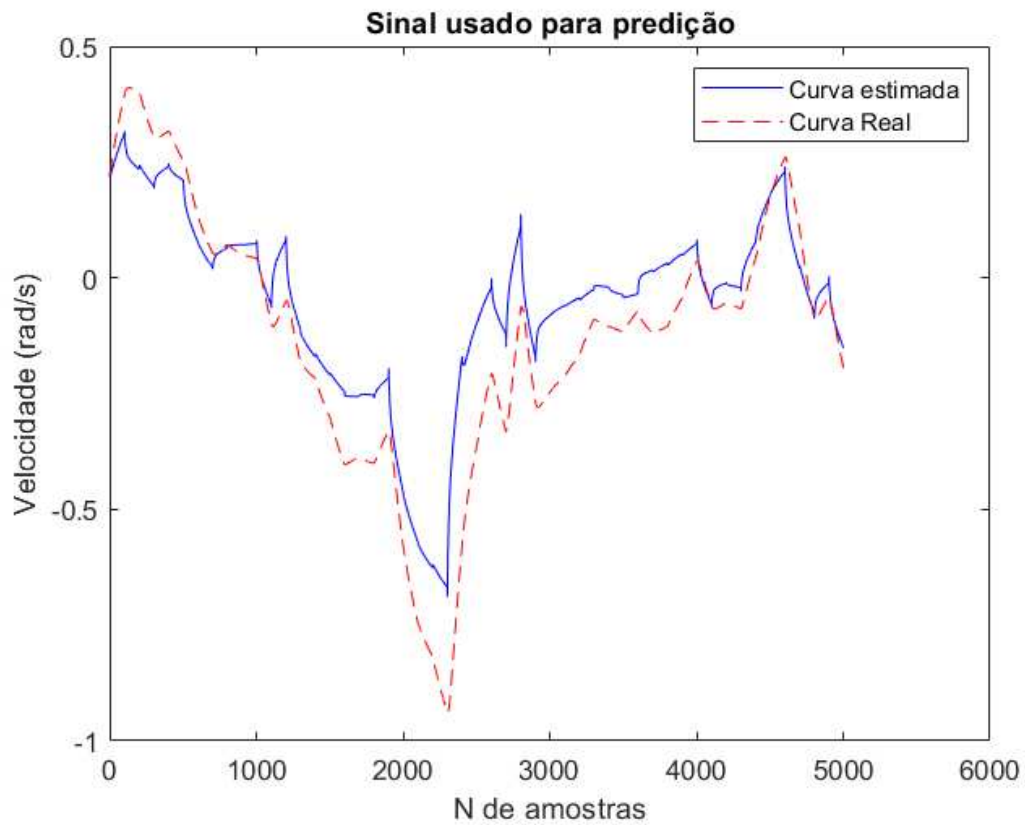
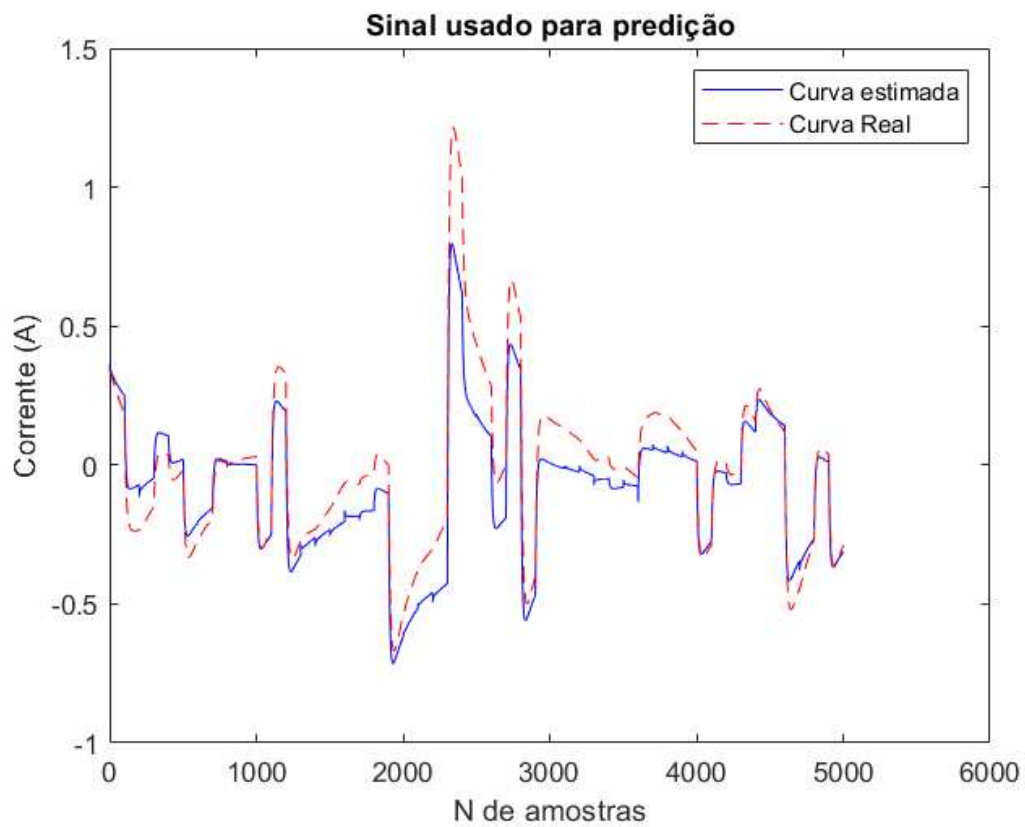


Figura 40 – Predição corrente teste 16



Assim como no teste 15, as observações são semelhantes. Na figura 39 que representa as curvas da velocidade é possível observar diferenças significativas entre a curva dos valores reais e os valores preditos e com erro médio quadrático de  $2.1797 \times 10^{-2}$ . Já na figura 40 que representa as curvas da corrente é menos perceptível essa diferença, mas ainda possível observar alguns erros que são representados numericamente pelo erro de  $1.8141 \times 10^{-2}$ .

### Teste 17

Ainda usando o teste 14 como base, o hiperparâmetro modificado dessa vez foi a função de ativação, agora sendo utilizada a função sigmóide e mantendo os outros valores.

- Número de neurônios na camada oculta = 25;
- Função de ativação = sigmoide;
- $P_{inicial} = 100000$ ;
- $W_{inicial} = 0.01$

Figura 41 – Predição velocidade teste 17

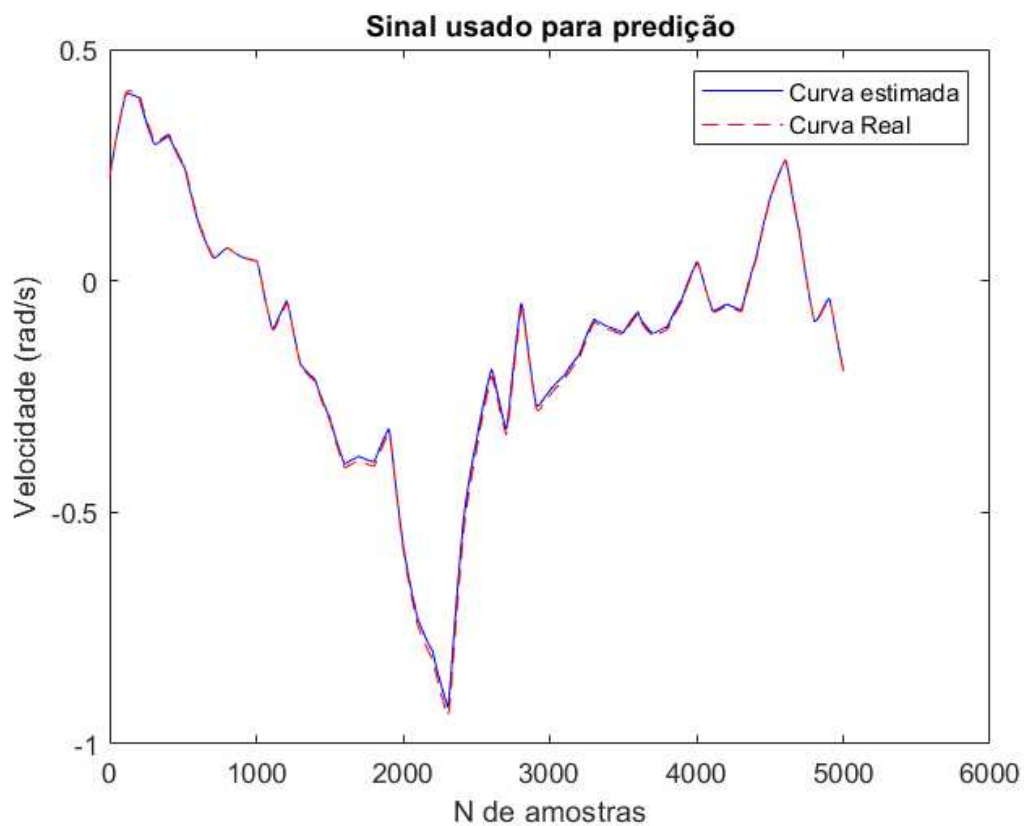
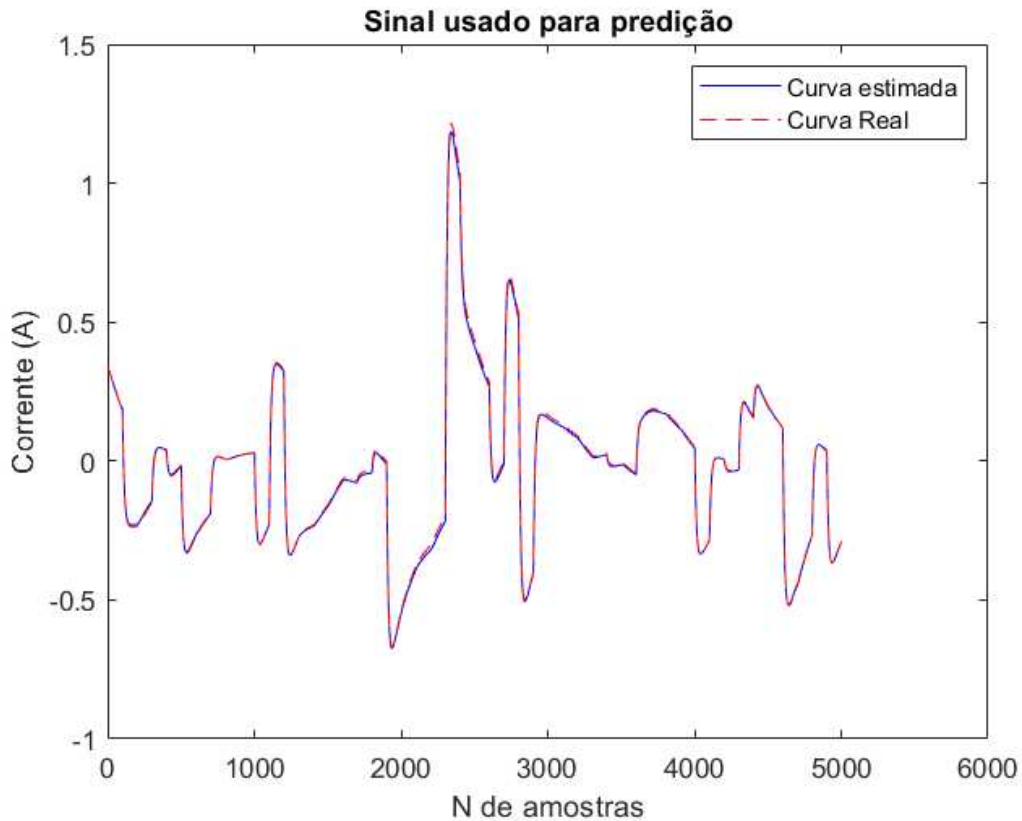




Figura 42 – Predição corrente teste 17



Nessa simulação um melhor resultado e poucas discrepâncias podem ser observadas visualmente, mostrando que, para essa tarefa, a mudança de função de ativação não piorou os resultados (como nas mudanças de  $W_{inicial}$  e  $P_{inicial}$ ). Com a métrica usada para quantificar isso, o MSE da figura 41 é de  $4.3306 \times 10^{-5}$  e o MSE da figura 42 é de  $4.0073 \times 10^{-5}$ .

### Teste 18

Para finalizar esse ciclo de simulações, e ainda usando o teste 14 como base, agora foi alterado o número de neurônios na camada oculta e mantidos os valores de  $P_{inicial}$  e  $W_{inicial}$  e função de ativação.

- Número de neurônios na camada oculta = 50;
- Função de ativação = tangente hiperbólica;
- $P_{inicial} = 100000$ ;
- $W_{inicial} = 0.01$

Figura 43 – Predição velocidade teste 18

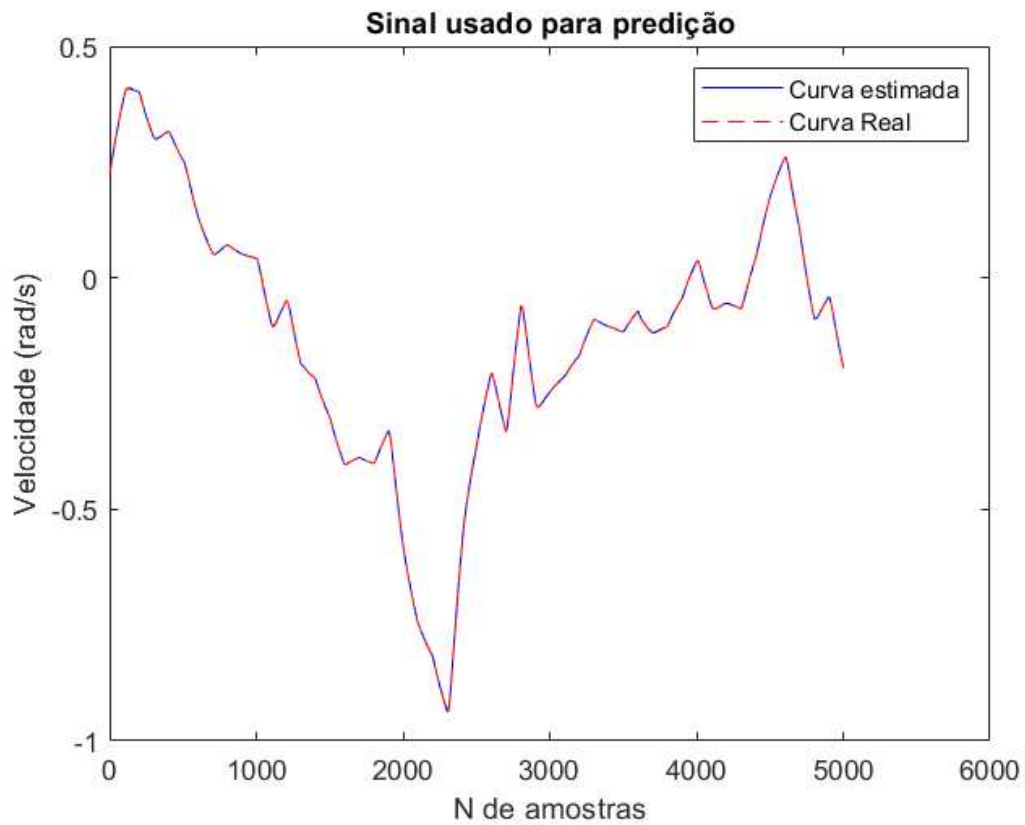
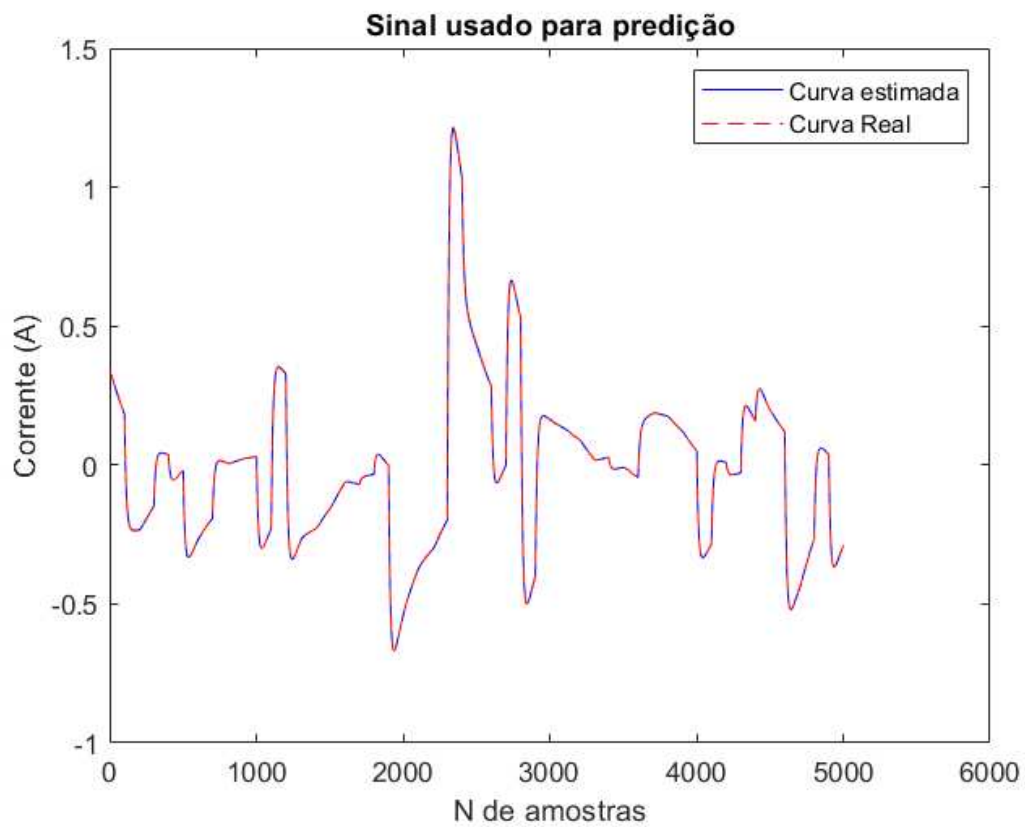


Figura 44 – Predição corrente teste 18



Na figura 43 o erro foi de  $1.7688 \times 10^{-6}$  e na figura 44 o erro foi de  $1.8720 \times 10^{-6}$ , representado uma qualidade muito boa da predição nesse teste com os hiperparâmetros selecionados.

A partir das observações dos testes 14 ao 18 e de acordo com os métodos descritos. Mesmo com o banco de dados e com o tipo de sinal de entrada diferente, os hiperparâmetros tem a mesma importância na qualidade dos resultados, o que pode ser descrito na tabela 12.

Tabela 12 – Resumo dos teste ELM Recursivo com entradas APRBS

Variável de saída	Teste 14	Teste 15	Teste 16	Teste 17	Teste 18
Velocidade	$1.4804 \times 10^{-5}$	$3.9907 \times 10^{-2}$	$2.1797 \times 10^{-2}$	$4.3306 \times 10^{-5}$	$1.7688 \times 10^{-6}$
Corrente	$1.3577 \times 10^{-5}$	$3.3916 \times 10^{-2}$	$1.8141 \times 10^{-2}$	$4.0073 \times 10^{-5}$	$1.8720 \times 10^{-6}$

A partir dos resultados das simulações realizadas, pode-se observar que os MSE tem ordem de grandeza semelhante nos dois bancos de dados.

Além disso, o ajuste dos hiperparâmetros  $W_{inicial}$  e  $P_{inicial}$  são fundamentais e sensíveis para a qualidade dos resultados. A ELM recursiva depende diretamente desses valores que são escolhidos por tentativa e erro (ou algum método de otimização de hiperparâmetros). Dentro do intervalo de valores para  $W_{inicial}$  e  $P_{inicial}$ , quanto maior os valores dentro do intervalo melhor a aproximação dos valores simulados com o banco de dados inicial.

E no teste 17 foi alterado o número de neurônios na camada oculta em relação ao teste 14 e houve uma diferença pouco significativa nos resultados. Para a ELM Recursiva, a importância dos pesos e do bias como descritos foram observados nos testes realizados. A escolha devida desses valores iniciais afetara diretamente na qualidade do treinamento e da consequente predição.

Para o banco de dados onde a tensão de entrada tensão e carga são sinais do tipo APRBS, os resultados das simulações estão resumidos na tabela a seguir:

Tabela 13 – Resumo dos testes com entradas APRBS

		ERRO QUADRÁTICO MÉDIO	
		VELOCIDADE	CORRENTE
<b>MLP</b>	TESTE 10	$6.5300 \times 10^{-2}$	$5.9904 \times 10^{-2}$
	TESTE 11	$1.1670 \times 10^{-1}$	$8.9782 \times 10^{-2}$
<b>ELM BATCH</b>	TESTE 12	$2.2339 \times 10^{-6}$	$3.1540 \times 10^{-5}$
	TESTE 13	$4.6309 \times 10^{-6}$	$1.3260 \times 10^{-5}$
<b>ELM ONLINE</b>	TESTE 14	$1.4804 \times 10^{-5}$	$1.3577 \times 10^{-5}$
	TESTE 15	$3.9907 \times 10^{-2}$	$3.3916 \times 10^{-2}$
	TESTE 16	$2.1797 \times 10^{-2}$	$1.8141 \times 10^{-2}$
	TESTE 17	$4.3306 \times 10^{-5}$	$4.0073 \times 10^{-5}$
	TESTE 18	$1.7688 \times 10^{-6}$	$1.8720 \times 10^{-6}$

Assim como nos teste com o banco de dados de entradas aleatórias, as conclusões em relação ao banco de dados com entrada APRBS são semelhantes. A ELM Recursiva se comporta da melhor maneira em comparação a ELM Não recursiva e a MLP. O que comprova a coerência das simulações com boa aplicação das técnicas apresentadas nesse trabalho.

#### 4.2.3 Resumo das simulações

Para catalogar todos os resultados e ter uma melhor visualização dos erros quadráticos médios de cada teste realizado a tabela abaixo apresenta todas as informações:

Tabela 14 – Resumo dos testes com entradas Aleatórias e APRBS

		ENTRADAS ALEATÓRIAS		ENTRADAS APRBS		
		ERRO QUADRÁTICO MÉDIO				
		VELOCIDADE	CORRENTE	VELOCIDADE	CORRENTE	
<b>MLP</b>	TESTE 01	1.7390e-02	5.8418e-03	6.5300e-02	5.9904e-02	TESTE 10
	TESTE 02	1.7463e-01	1.1483e-02	1.1670e-01	8.9782e-02	TESTE 11
<b>ELM BATCH</b>	TESTE 03	9.3116e-06	5.6083e-06	2.233e-06	3.1540e-05	TESTE 12
	TESTE 04	4.4956e-05	4.1675e-05	4.4956e-05	4.1675e-05	TESTE 13
<b>ELM ONLINE</b>	TESTE 05	2.9950e-06	1.6433e-07	2.9950e-06	1.6433e-07	TESTE 14
	TESTE 06	3.3377e-02	2.0795e-03	3.3377e-02	2.0795e-03	TESTE 15
	TESTE 07	2.684e-02	2.067e-03	2.684e-02	2.067e-03	TESTE 16
	TESTE 08	2.9156e-04	2.0477e-05	2.9156e-04	2.0477e-05	TESTE 17
	TESTE 09	2.5733e-06	1.0255e-07	2.5733e-06	1.0255e-07	TESTE 18

Para os testes da MLP dos dois bancos de dados apresentados não há diferença na ordem de grandeza dos erros, e com erros relativamente altos o que demonstra a baixa qualidade da MLP para a predição do sistema apresentado no trabalho. Além do algoritmo de treinamento

da  $ELM_b$  ter um custo computacional menor do que o backpropagation, este levou a resultado melhores que o da MLP. Por fim, a  $ELM_o$  obteve os melhores resultados, porém esta é mais dependente da otimização de alguns hiperparâmetros.

## 5 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho, buscou-se comparar diferentes métodos de treinamento de redes neurais artificiais do tipo perceptron multicamadas, com uma única camada oculta. A partir da simulação de um motor CC sujeito a diferentes valores de carga e tensão, foram gerados dois bancos de dados, cada um composto de 4 sinais (carga, tensão, corrente e velocidade) de 10006 amostras cada. Os hiperparâmetros dos métodos de treinamento BP,  $ELM_b$  e  $ELM_o$  foram variados, e a influência destes na medida de MSE entre os valores reais e preditos foi verificada. Em geral, a MLP treinada pelo método MP obteve os piores resultados de MSE. Já a  $ELM_o$  teve os menores valores MSE. Os resultados dos dois bancos de dados tiveram poucas diferenças, não apresentando evidências de qual seria o melhor tipo de sinal de excitação (aleatório ou APRBS) a ser aplicado no sistema em questão.

Os objetivos específicos deste trabalho foram alcançados, já que foram gerados dois bancos de dados a partir de um sistema simulado, diferentes algoritmos de treinamento de redes neurais artificiais foram implementados no software Matlab, e foi verificada a influência da alteração dos hiperparâmetros destes algoritmos no MSE entre os sinais reais e estimados.

Como principais trabalhos futuros, podem ser citados:

- Geração de banco de dados a partir de testes experimentais em bancada;
- Utilizar métodos de otimização para a definição dos hiperparâmetros para a construção de modelos na tarefa de identificação de sistemas;
- Utilizar o método de treinamento da rede ELM de modo a projetar controladores para os sistemas estimados;
- Embarcar os controladores projetados em um microcontrolador.

Além destes, mais bancos de dados podem ser utilizados para validar a identificação através das redes neurais implementadas e podem ser gerados novos bancos de dados com outros tipos de sinais de excitação. Por fim, os sinais de tensão e corrente dos bancos de dados já gerados podem ser utilizados como entradas de modo a estimar a velocidade e a carga do motor.

## REFERÊNCIAS

- AGUIRRE, L. **Introdução à Identificação de Sistemas**. [S.l.: s.n.], 2015. ISBN 978-85-423-0079-6.
- ALBADR, M.; TIUNA, S. Extreme learning machine: A review. **International Journal of Applied Engineering Research**, v. 12, p. 4610–4623, 01 2017.
- BESSA, R.; BARRETO, G. Robust echo state network for recursive system identification. In: \_\_\_\_\_. [S.l.: s.n.], 2019. p. 247–258. ISBN 978-3-030-20520-1.
- BILLINGS, S. A. Nonlinear system identification: Narmax methods in the time, frequency, and spatio-temporal domains. In: . [S.l.: s.n.], 2013.
- CHIUSO, A.; PILLONETTO, G. System identification: A machine learning perspective. **Annual Review of Control, Robotics, and Autonomous Systems**, v. 2, n. 1, p. 281–304, 2019.
- COELHO, A.; COELHO, L. **Identificação de Sistemas Dinâmicos**. [S.l.: s.n.], 2004. ISBN 978.85.328.0730-4.
- DINARY, M. **A Control Scheme for Industrial Robots Using Artificial Neural Networks**. Tese (Doutorado), 02 2015.
- FORGIONE, M.; PIGA, D. Continuous-time system identification with neural networks: Model structures and fitting criteria. **European Journal of Control**, v. 59, 02 2021.
- GREGORÄIÄ, G.; LIGHTBODY, G. Nonlinear system identification: From multiple-model networks to gaussian processes. **Engineering Applications of Artificial Intelligence**, v. 21, n. 7, p. 1035–1055, 2008. ISSN 0952-1976. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0952197607001467>>.
- GRETTON, A.; DOUCET, A.; HERBRICH, R.; RAYNER, P.; SCHOLKOPF, B. Support vector regression for black-box system identification. In: **Proceedings of the 11th IEEE Signal Processing Workshop on Statistical Signal Processing (Cat. No.01TH8563)**. [S.l.: s.n.], 2001. p. 341–344.
- HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. **Neural Networks**, v. 2, n. 5, p. 359–366, 1989. ISSN 0893-6080. Disponível em: <<https://www.sciencedirect.com/science/article/pii/0893608089900208>>.
- HUANG, G.-B.; LIANG, N.; RONG, H.-J.; SARATCHANDRAN, P.; SUNDARARAJAN, N. On-line sequential extreme learning machine. In: . [S.l.: s.n.], 2005. v. 2005, p. 232–237.
- HUANG, G.-B.; WANG, D.; LAN, Y. Extreme learning machines: A survey. **IJMLC**, p. 1–16, 01 2011.
- HUANG, G.-B.; ZHU, Q.-Y.; SIEW, C. Extreme learning machine: A new learning scheme of feedforward neural networks. In: . [S.l.: s.n.], 2004. v. 2, p. 985 – 990 vol.2. ISBN 0-7803-8359-1.
- ISERMANN, R.; MÜNCHHOF, M. Identification of dynamic systems: An introduction with applications. In: . [S.l.: s.n.], 2010.

KUMAR, D.-R.; SRIVASTAVA, S. Comparative study of neural networks for dynamic nonlinear systems identification. **Soft Computing**, v. 23, 01 2019.

LI, Y.; QIU, R.; JING, S. Intrusion detection system using online sequence extreme learning machine (os-elm) in advanced metering infrastructure of smart grid. **PLOS ONE**, Public Library of Science, v. 13, n. 2, p. 1–16, 02 2018. Disponível em: <<https://doi.org/10.1371/journal.pone.0192216>>.

MARTINEZ-RAMON, M.; ROJO-ALVAREZ, J. L.; CAMPS-VALLS, G.; MUNOZ-MARI, J.; NAVIA-VAZQUEZ, n.; SORIA-OLIVAS, E.; FIGUEIRAS-VIDAL, A. R. Support vector machines for nonlinear kernel arma system identification. **IEEE Transactions on Neural Networks**, v. 17, n. 6, p. 1617–1622, 2006.

MASHOR, M. Performance comparison between hmlp, mlp and rbf networks with application to on-line system identification. In: **IEEE Conference on Cybernetics and Intelligent Systems**, 2004. [S.l.: s.n.], 2004. v. 1, p. 643–648 vol.1.

PATTANAIK, R. K.; MOHANTY, M. N.; MOHAPATRA, S. K.; KU, B. . P. Nonlinear dynamic system identification of arx model for speech signal identification. **Computer Systems Science and Engineering**, v. 46, n. 1, p. 195–208, 2023. Disponível em: <<http://www.techscience.com/csse/v46n1/51291>>.

PRASAD, V.; BEQUETTE, B. Nonlinear system identification and model reduction using artificial neural networks. **Computers and Chemical Engineering**, v. 27, n. 12, p. 1741–1754, 2003. ISSN 0098-1354. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0098135403001376>>.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, v. 65 6, p. 386–408, 1958.

ROWELL, D. Introduction to recursive-least-squares (rls) adaptive filters. 2006.

SILVA, I. Nunes da; SPATTI, D. H.; ANDRADE, R. **Redes Neurais Artificiais Para Engenharia e Ciências Aplicadas**. [S.l.: s.n.], 2016.

TIWARI, S.; NARESH, R.; JHA, R. Comparative study of backpropagation algorithms in neural network based identification of power system. **International Journal of Computer Science & Information Technology**, Academy & Industry Research Collaboration Center (AIRCC), v. 5, n. 4, p. 93, 2013.

WANG, J.; LU, S. A review on extreme learning machine. 05 2021.

WANG, J.; LU, S.; WANG, S.-H.; ZHANG, Y.-D. A review on extreme learning machine. In: . [S.l.: s.n.], 2022.

WEI, Y.; XIA, L.; PAN, S.; WU, J.; ZHANG, X.; HAN, M.; ZHANG, W.; XIE, J.; LI, Q. Prediction of occupancy level and energy consumption in office building using blind system identification and neural networks. **Applied Energy**, v. 240, p. 276–294, 2019. ISSN 0306-2619. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0306261919303538>>.

WERBOS, P.; JOHN, P. Beyond regression : new tools for prediction and analysis in the behavioral sciences /. 01 1974.



YE, Y.; SQUARTINI, S.; PIAZZA, F. Online sequential extreme learning machine in nonstationary environments. **Neurocomputing**, v. 116, p. 94–101, 2013. ISSN 0925-2312. Advanced Theory and Methodology in Intelligent Computing. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S092523121200728X>>.

**APÊNDICE A – ALGORITMO DE GERAÇÃO DO SINAL APRBS**

---

**Algoritmo 1:** Algoritmo de geração de sinal APRBS através de um sinal PRBS noSimulink

---

```
function y = fcn(u)
persistent last input;
persistent last output;
if isempty(last input)
last input = u;
last output = 100*randn;
else
if u = last input
last input = u;
last output = 100*randn;
end
end
y = last output;
```

---