ESCALONAMENTO DE GANHOS EM CONTROLADORES PID/PI COM A UTILIZAÇÃO DE REDES NEURAIS ARTIFICIAIS APLICADOS EM MÁQUINAS DE EMPACOTAMENTO

Fabio Bruci Wu. E-mail: fabio.wu@hotmail.com
Paulo Roberto Barbosa. E-mail: paulorb@ifsp.edu.br

Resumo: Sistemas controlados por servo motor são facilmente encontrados na indústria e, na maioria dos casos, os sistemas não são lineares, o que pode demandar o desenvolvimento de estratégias mais elaboradas para o controle do servo motor. A planta utilizada neste estudo é um conjunto de selagem acionado por servo motor onde o controle é feito diretamente por um servo *drive* com controle PID/PI (Proporcional Integral Derivativo/Proporcional Integral), onde é possível manipular os valores dos ganhos do controlador. O conjunto de selagem efetua o movimento de abertura e fechamento de forma cíclica, produzindo um sobressinal de posição de 2% ou mais quando ajustado pelo método de sintonização automática dos ganhos. Sabendo que a planta não é linear e que possui diferentes ganhos para diferentes faixas de operação, este trabalho tem como objetivo propor um escalonamento de ganhos para as diferentes faixas de operação utilizando uma rede neural artificial capaz de reproduzir uma função que irá gerar os ganhos em função da faixa de operação do sistema, reduzindo o sobressinal do conjunto de 2% ou mais para um sobressinal menor que 0,5%.

Palavras-chave: Neural gain scheduling, gain scheduling, controle de servo motor, rede neural artificial.

GAIN SCHEDULING IN PID/PI CONTROLLERS USING ARTIFICIAL NEURAL NETWORKS APPLIED IN PACKING MACHINES

Abstract: Servo motor controlled systems are readily available in the industry and in most cases, the systems are nonlinear which may require developing more elaborate strategies for engine control. The plant used in this study is a set of servo motor driven sealing where control is done directly by a servo drive with PID/PI control (proportional integral derivative/proportional integral) where you can manipulate the values of the controller gains. The sealing assembly performs the opening and closing movement cyclically, producing a position overhangs of 2% or more when adjusted by the automatic gain tuning method. Knowing that the plant is not linear and that has different gains for different operating ranges, this paper aims to propose a gain schedule for the different operating ranges using an artificial neural network capable of playing a function that will generate the gains according to the system operating range, improving control performance throughout its operating range, Reducing the salience of the set of 2% or more to a salient of less than 0.5%.

Keywords: Neural gain schedule, gain schedule, servo motor control, artificial neural network.

1. INTRODUÇÃO

O segmento de máquinas de empacotamento para o setor alimentício é um setor da indústria que possui forte automação, composta por atuadores pneumáticos, servo motores, motores trifásicos, encoder, etc. Uma máquina de empacotamento vertical pode fabricar pacotes do tipo almofada e necessita de um controle preciso para atingir uma boa velocidade de produção e obter uma durabilidade alta.

Um conjunto fundamental em uma máquina de empacotamento vertical é o mordente de selagem horizontal, responsável por fazer a selagem horizontal de pacotes, controlando as variáveis de pressão, torque de selagem, temperatura e tempo de selagem. O objetivo é obter uma selagem sem rugas, estrias e não deixar que o conteúdo do pacote entre em contato com o meio externo, portanto na selagem não pode haver vazamentos.

O conjunto de selagem, o qual será analisado, é acionado por um servo motor controlado por um servo *drive* e um CLP (Controlador Lógico Programável), todos de uso comum na indústria. Este possui uma construção mecânica que torna o sistema não linear de forma a dificultar o controle do servo motor.

É comum na indústria de máquinas a utilização de controladores que utilizam um controle do tipo PID (Proporcional Integral Derivativo) para controlar servo motores e, em poucos casos, é realizada a modelagem do sistema para calcular os ganhos do controlador. Uma forma de não precisar calcular os ganhos é fazer o procedimento de sintonização automática (autotuning), diretamente no *drive*. O sistema automático efetua um movimento de acordo com uma configuração feita pelo usuário, e de acordo com a resposta do sistema, efetua o cálculo dos ganhos. Esse procedimento é muito eficaz na grande maioria dos casos, porém em plantas onde a não linearidade é muito grande, se torna difícil encontrar um único conjunto de ganhos para toda a faixa de operação do sistema.

Visando corrigir a questão de o controlador possuir diferentes ganhos para diferentes faixas de operação do sistema, este artigo sugere uma forma de escalonar os ganhos ajustados para diferentes faixas de operação através de uma rede neural artificial, com o objetivo de reduzir o sobressinal de posicionamento na abertura dos mordentes de 2% para 0,5%.

2. DINÂMICA DO CONJUNTO

Conforme ilustrado na Figura 1, a rotação do servo motor faz com que o sistema de barras desloque os mordentes da posição de abertura para a posição de fechamento de forma cíclica, com uma frequência máxima de 60 ciclos por minuto, sendo a posição de abertura determinada em 120mm de abertura e a posição de fechamento quando os mordentes se tocam.

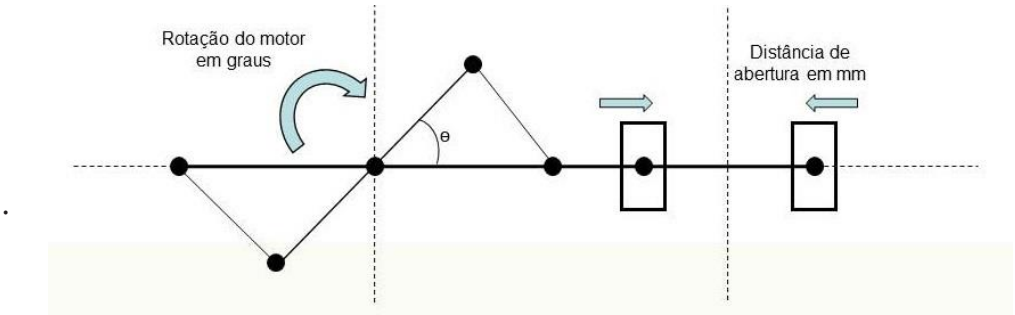

Figura 1 – Esquema do Mordente – Posição de Abertura.

Quando os mordentes retornam para a posição de abertura, ocorre um sobressinal de posicionamento de cerca de 2%, quando feito o ajuste automático de parâmetros, este sobressinal acarreta em diminuição da vida útilil de componentes mecânicos além de demonstrar a falha de controle sobre o sistema.

A rotação do servo motor é controlada diretamente por um servo *drive* do tipo Kinetix300 Ethernet/IP fabricado pela Allen Bradley, que controla o servo motor através de index de posicionamento, que são preenchidos através de comandos enviados por um CLP. O servo *drive* possui uma malha de controle fechada com um controlador PID (Proporcional Integral Derivativo) de posição e um controlador PI (Proporcional Integral) de velocidade, onde os ganhos podem ser escritos pelos comandos do CLP.

3. REDES NEURAIS ARTIFICIAIS

De acordo com Haykin (2001), o estudo de redes neurais artificiais (RNA) tem sido motivado pelo reconhecimento da grande capacidade de processamento de informações do cérebro humano, que é complexo, não linear e processa informações paralelamente, diferente de um computador digital convencional. "Na sua forma mais geral, uma rede neural é uma máquina que é projetada para modelar a maneira que o cérebro realiza uma tarefa particular ou função de interesse; Ou seja, pode-se dizer que uma RNA é uma tentativa de simular a forma como o cérebro realiza uma tarefa; a rede é normalmente implementada utilizando-se componentes eletrônicos ou é simulada por programação em um computador digital"(HAYKIN,2001,p.28).

3.1 Perceptron de camada única

Segundo Haykin (2001), um neurônio artificial é uma unidade de processamento de informações básica e fundamental para a operação de uma rede neural artificial.

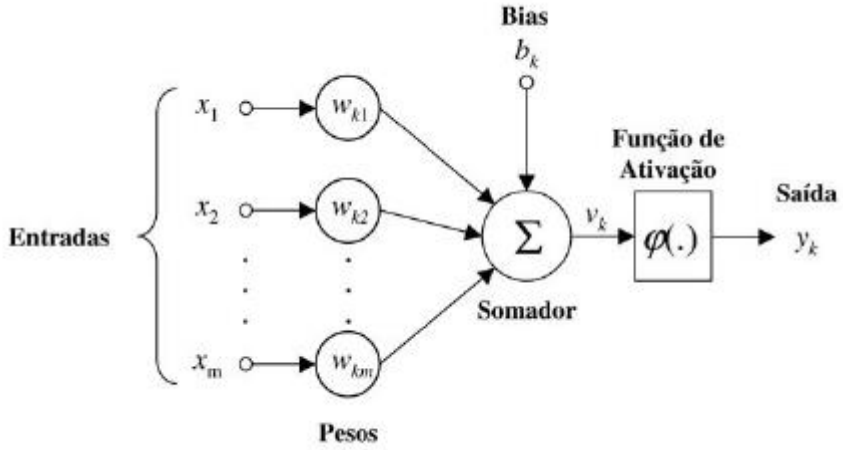

Figura 2 – Modelo matemático de um neurônio (HAYKIN, 2001).

Na Figura 2 é apresentado um modelo de neurônio onde:

- $X_1, X_2 \ldots X_m$ são as entradas do sistema que serão utilizadas no controle e no treinamento da rede - é por onde o neurônio recebe a informação.
- $W_{k1}, W_{k2} \ldots W_{km}$ são os pesos sinápticos, os quais são multiplicados pelo valor da entrada ponderando cada uma das entradas, o intuito do treinamento da rede é encontrar o valor ideal para cada peso sináptico para que a saída seja a desejada.
- O bias b_k é uma entrada externa que tem como função aumentar ou diminuir o valor no somador. Bias é um valor fixo que pode variar de acordo com a aplicação do neurônio e pode ser visto como uma entrada X_0 multiplicada por um peso W_0.

- O somador tem a função de somar as entradas já ponderadas dos seus respectivos pesos.
- A função de ativação (φ), também referida como função restritiva, restringe a amplitude da saída de um neurônio. Existem diferentes tipos de função de ativação que irão variar diretamente com a aplicação de cada neurônio.

3.2 Função de ativação Sigmoide

É uma forma bastante comum na construção de redes neurais artificiais, possui forma de S como pode-se observar na Figura 3. Por definição é uma função estritamente crescente que exibe um balanceamento adequado entre comportamento linear e não linear. Um exemplo de função sigmoide é a função logística matematicamente representada pela equação 1, onde a é o parâmetro de inclinação da função sigmoide. É possível variar a inclinação da função sigmoide variando esse parâmetro.

$$\varphi(v) = \frac{1}{1 + \exp(-av)} \tag{1}$$

A função de ativação definida na equação 1 se estende no intervalo de 0 a +1, porém, ela também podem se estender de -1 a +1, tornando-se antissimétrica em relação a origem. A equação 2 é chamada de função sinal.

$$\varphi(\mathcal{V}) = \begin{cases} 1 & \text{se } \mathcal{V} > 0 \\ 0 & \text{se } \mathcal{V} = 0 \\ -1 & \text{se } \mathcal{V} < 0 \end{cases} \tag{2}$$

Para a forma correspondente de uma função sigmoide, pode-se utilizar a função tangente hiperbólica, descrita pela equação 3.

$$\varphi(\mathcal{V}) = \tanh(\mathcal{V}) \tag{3}$$

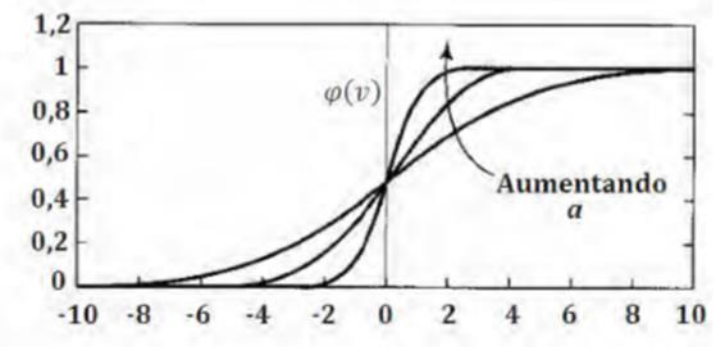

Figura 3 – Função Sigmoide (HAYKIN, 2001).

3.3 Perceptron de múltiplas camadas

O *Perceptron* de Múltiplas Camadas (MLP, *multilayer perceptron*), é aplicado a problemas mais complexos que não podem ser solucionados com o *perceptron* de camada única. Possuem além da camada de entrada e de saída, uma ou mais camadas ocultas, que irão variar de acordo com o tipo de problema a ser solucionado.

Na Figura 4, pode-se observar uma rede neural artificial do tipo *perceptron* de múltiplas camadas, que é composta por uma camada de entrada, uma camada de saída e duas camadas ocultas. Essa rede é chamada de totalmente conectada, pois cada neurônio em qualquer camada está conectado a cada um dos neurônios da camada anterior. O sinal de entrada (estímulo) é propagado para frente, da esquerda para a direita, pela rede, por suas conexões sinápticas.

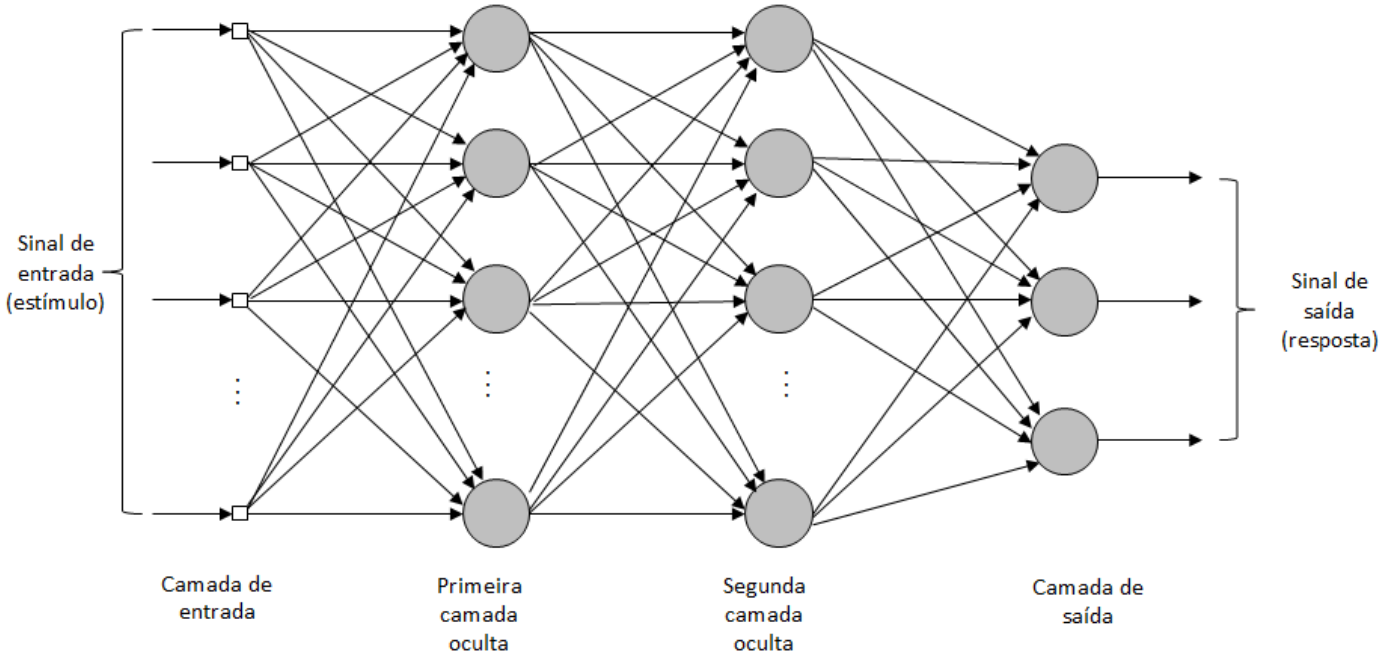

Figura 4 – Arquitetura de uma PMC com duas camadas ocultas (HAYKIN, 2001).

O treinamento desta RNA, ou, o ajuste dos pesos sinápticos, é feito pelo algoritmo de retropropagação do erro (*error back-propagation*). Este tipo de treinamento é composto, basicamente, por dois passos:

1 - A propagação, que consiste em aplicar um vetor de entrada a camada de entrada, sendo propagado pela rede até gerar uma saída. A saída ideal é subtraída da saída real, gerando um sinal de erro. Os pesos sinápticos se mantêm fixos nesta etapa.

2 - A retropropagação, que consiste em propagar o sinal de erro no sentido contrário das conexões sinápticas, ajustando os pesos sinápticos para que a saída real seja mais próxima da saída desejada.

O algoritmo para treinar os neurônios em uma rede multicamadas é um tanto complexo e difícil de visualizar. Para exemplificar matematicamente o algoritmo de treinamento deve-se observar se o neurônio em questão está na camada oculta ou se está na camada de saída.

As amostras são apresentadas à rede neural artificial, que pode ser realizado de duas formas: modo sequencial, ou seja, é fornecido uma amostra a rede e após os passos de propagação e retropropagação os pesos são alterados e, assim, sucessivamente, a cada nova amostra de dados, ou por lote, apresentando um conjunto de dados e, somente após todos os dados serem apresentados a rede, os pesos são ajustados.

4. AQUISIÇÃO DE DADOS

Para que a rede neural artificial possa ser treinada é necessário que os dados de treinamento sejam adquiridos, desta forma, as diferentes faixas de operação do servo motor serão delimitadas e para cada região será ajustado um conjunto de ganhos no controlador.

4.1 Determinação das faixas de operação

Para que o escalonamento de ganhos seja feito em diferentes faixas de operação do sistema é necessário identificar essas regiões. Na planta em estudo, foi observado que a variação da posição do servo motor não é linear com o deslocamento da massa do conjunto.

Para observar essa não linearidade foi feito um experimento incrementando um valor em unidades do servo motor, que é proporcional a graus de rotação, e medido o deslocamento linear da região onde se encontra o mordente, a origem da medição foi com o mordente fechado, onde foi considerado como deslocamento zero. Foi obtido um gráfico não linear conforme ilustrado no Gráfico 1.

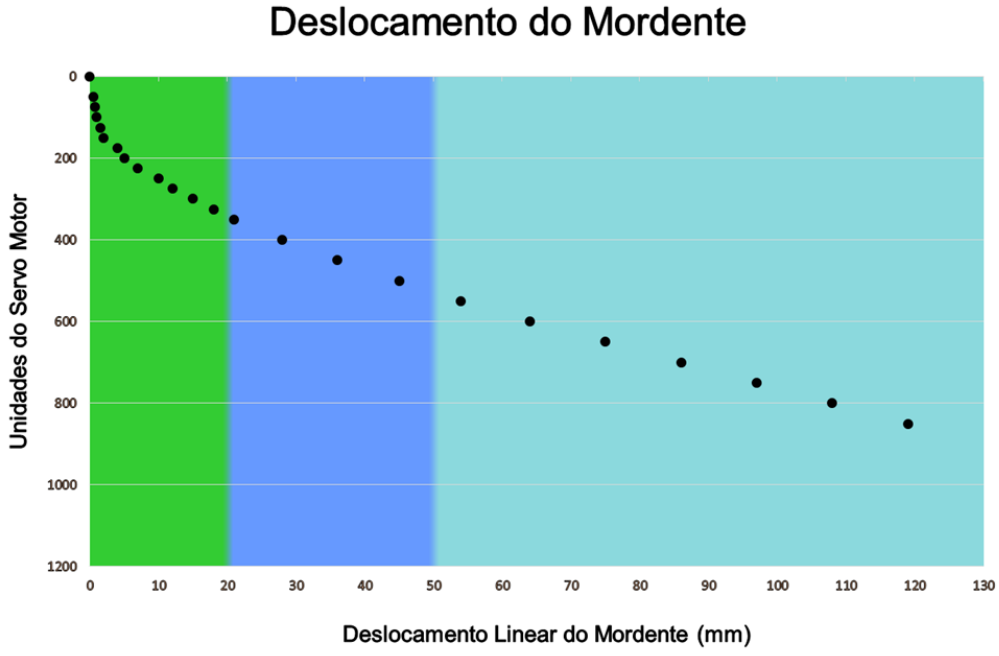

Gráfico 1 – Deslocamento do mordente.

De acordo com o Gráfico 1, pode-se observar que o movimento se inicia de forma não linear e, posteriormente, se torna linear. Desta forma, o movimento foi dividido em três regiões (A, B e C), que serão consideradas diferentes faixas de operação do controlador, ou seja, cada região terá seus ganhos ajustados para quando o posicionamento do servo motor estiver dentro deste limite, considerando a região A, quando o servo motor estiver entre a posição 0 e a posição 20 (mm), a região B, quando o servo motor estiver entre a posição 20 e 50 (mm) e a região C, quando o servo motor estiver entre a posição 50 e 120 (mm).

4.2 Ajuste de ganhos do controlador

A condição de ajuste inicial do controlador foi feita de forma empírica partindo do ajuste automático do servo motor, até obter ganhos que demonstrassem um controle aceitável. Pode-se observar pela Figura 5, a curva de posição do servo motor durante o movimento de abertura e fechamento, onde pode ser observado o sobressinal de posicionamento na posição de abertura de 2%, a posição de fechamento é o trecho retilíneo localizado na parte superior da curva.

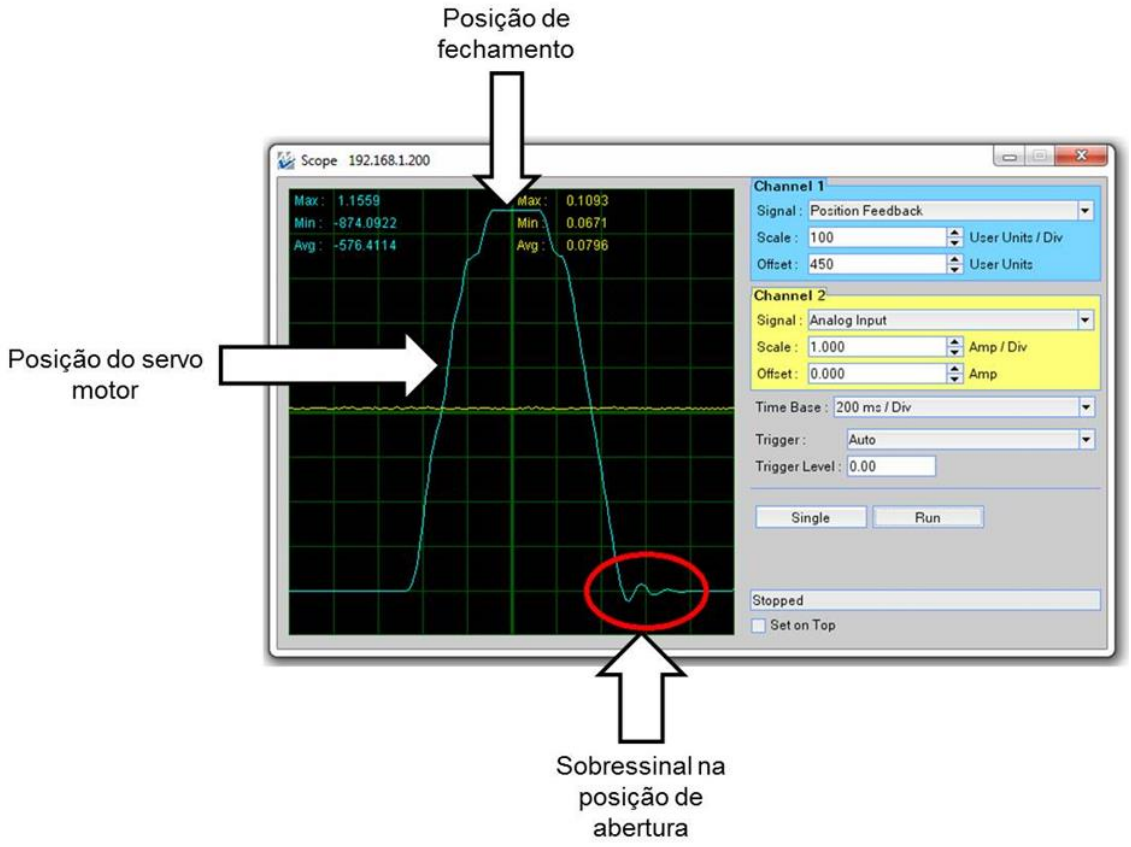

Figura 5 – Curva do servo motor com ajustes iniciais.

Para ajustar os ganhos da região A, foi parametrizado no servo *drive* a posição de fechamento 0mm (ponto onde os mordentes se encontram) e a de abertura 20mm, fazendo o servo motor se deslocar da posição de abertura para a posição de fechamento e vice-versa, de forma cíclica, na faixa de operação da região A, desta forma os ganhos foram ajustados com o conjunto em movimento, observando a redução do sobressinal na posição de abertura.

Para a região A os ganhos foram mantidos com o objetivo de registrar o erro que pode ser observado na Figura 6 (lado esquerdo) e posteriormente os ganhos foram ajustados, obtendo um conjunto de 20 ganhos diferentes para a região A, de forma que o sobressinal de posicionamento fosse menor que 0,5%, que pode ser observado na Figura 6 (lado direito).

Figura 6 – Sobressinal de posição com ganhos ajustados para a regiâo A.

Desta forma, as três faixas de operação do conjunto foram ajustadas obtendo-se três conjuntos de 20 diferentes ganhos, para serem utilizados no treinamento, teste e validação da rede neural artificial.

5. RNA: DEFINIÇÃO, TREINAMENTO E SIMULAÇÃO

Para que a rede neural artificial seja projetada é necessário definir de forma clara suas configurações como definição de entradas e saídas, organização dos dados para treinamento, arquitetura da rede, para que enfim a rede possa ser simulada no programa computacional Matlab.

5.1 Definições das entradas e saídas

Para que a RNA seja projetada, é necessário que as entradas e saídas da rede neural artificial estejam definidas, bem como os limites que possam existir entre valores de entrada e saída.

A posição do servomotor do conjunto do mordente de selagem é limitada fisicamente no espaço amostral que varia de 0 até 120mm de abertura. A posição do servomotor será a variável de entrada da rede neural artificial, pois é ela quem representa a mudança da faixa de operação da planta.

O controlador possui 5 ganhos que serão escalonados, KP, KI e KD de posição e KP e KI de velocidade. Estes ganhos serão as variáveis de saída da rede neural artificial e o seu limite se encontra através do valor máximo e mínimo coletados, portanto a RNA terá uma entrada e cinco saídas.

Utilizando-se os ganhos coletados para cada região, e todo o espaço amostral possível das entradas (0 – 120mm), foi criada uma tabela para cada região de operação do controlador que será utilizada no treinamento, teste e validação da rede no programa computacional Matlab.

A região A possui 20 diferentes dados de entrada (0 – 19). Foram coletados 20 diferentes conjuntos de ganhos e foi ordenado de forma randômica um conjunto de dados para cada posição do servo motor da faixa de operação A.

A região B possui 30 diferentes dados de entrada (20 – 50). Foram coletados 20 diferentes conjuntos de ganhos. Os conjuntos de ganhos coletados foram distribuídos de forma randômica para as posições do servo motor da faixa de operação B.

A região C possui 70 diferentes dados de entrada (50 – 120). Foram coletados 20 diferentes conjuntos de ganhos. Os conjuntos de ganhos coletados foram distribuídos de forma randômica para as posições do servo motor da faixa de operação C.

5.2 Treinamento da RNA

A partir dos dados definidos anteriormente a rede foi configurada e treinada para atingir o objetivo de escalonar os ganhos do controlador nas diferentes faixas de operação.

As entradas e saídas já estão definidas conforme visto, uma entrada e cinco saídas. Segundo Miguez (2012), não há uma regra exata para a determinação da quantidade de camadas ocultas, porém, existem estudos que fazem aproximações, segundo Hecht-Nielsen(1989), ao utilizar a retropropagação do erro, apenas uma camada oculta é necessária para aproximar qualquer função. Desta forma a rede neural artificial foi configurada com apenas uma camada oculta.

Segundo Binoti et al. (2014), a quantidade de neurônios na camada oculta é determinada principalmente de maneira empírica, pela experiência de quem projeta a rede neural artificial, embora existam estudos propostos como por Hirose et al. (1991), Arai (1993) e Fujita (1998). Hecht-Nielsen(1987), determinou que em uma rede neural com retropropagação do erro, onde possui apenas uma camada oculta, dever possui 2n+1 neurônios na camada oculta, onde n é a quantidade de entradas da rede neural artificial, portanto no caso da RNA para o escalonamento dos ganhos do mordente horizontal seriam necessários 3 neurônios na camada oculta.

Na determinação do número de neurônios deve ser considerado que um número excessivo de neurônios pode acarretar na memorização dos dados de treinamento, também conhecido como *overfiting*. Porém, um número pequeno de neurônios na camada oculta pode não atingir o resultado esperado da rede neural artificial, este processo é conhecido como *underfiting* (SILVA et al. 2010).

Desta forma, a rede neural artificial foi inicialmente configurada com 3 neurônios na camada oculta, porém, após testes, foi constatado que com apenas 2 neurônios na camada oculta, seria possível treinar a rede neural artificial de forma que ela obtivesse o desempenho esperado. A configuração final da RNA pode ser vista na Figura 7.

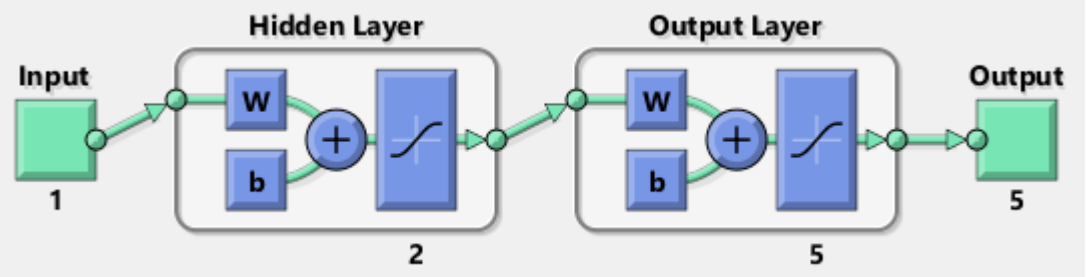

Figura 7 – Arquitetura da rede neural artificial.

A função de ativação utilizada foi a função descrita no Matlab como TANSIG que segundo o site mathworks do Matlab, é a função de transferência sigmoide tangente hiperbólica. Foram testadas outras funções, porém esta foi a que demonstrou melhor desempenho para o projeto.

A função de treinamento utilizada foi a função descrita no Matlab como TRAINBR, que segundo o site mathworks do Matlab é a função de regularização bayesiana que consiste em atualizar os valores dos pesos e do bias de acordo com a otimização de Levenberg-Marquardt. Isso minimiza a combinação do erro quadrático e dos pesos, e determina a combinação correta para produzir uma rede que faça a generalização.

A função de performance utilizada foi a função MSE, que segundo o site mathworks do Matlab é o erro médio quadrático. Esta função de performance deve ser utilizada quando a função de treinamento utiliza a regularização bayesiana.

Foram feitos diversos testes com funções de treinamento diferentes e diferentes configurações, porém, esta foi a configuração que obteve melhor resultado com um menor número de neurônios. Pode ser visto na Figura 8 a curva de desempenho do treinamento onde é mostrado o decaimento do erro no programa computacional Matlab.

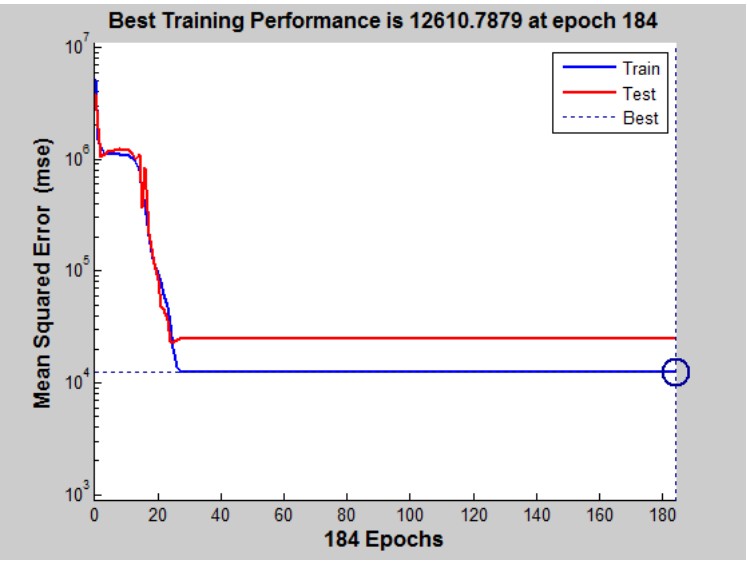

Figura 8 – Desempenho do treinamento.

Segundo Beale (2016), no treinamento de redes multicamadas é usual que os dados sejam divididos em três conjuntos. O primeiro é o de treinamento, utilizado para computar o gradiente e atualizar os valores de peso e bias. O segundo é o de validação, onde o erro é monitorado no processo do treinamento. O terceiro conjunto é o de teste, que não é utilizado no treinamento, mas sim para comparar diferentes modelos e também para plotar o erro do conjunto de teste durante o treinamento. A proporção da divisão dos dados utilizada no conjunto do mordente foi a padrão do Matlab, sendo 70% dos dados utilizados no treinamento, 15% na validação e 15% no teste, sendo que a definição de quais dados fazem parte de quais conjuntos foi feita de modo aleatório, também padrão do Matlab.O treinamento parou pelo "Mu" máximo exibindo a mensagem "Maximum Mu reached", segundo Abhishek (2012), Mu é um parâmetro que mede taxa de adaptação/aprendizagem, a parada do treinamento pelo Mu máximo significa que a taxa de aprendizagem atingiu o seu máximo, que o aprendizado da rede não irá mais evoluir, por este motivo o treinamento é encerrado.

6. APLICAÇÃO DA RNA NO CLP

Após ser treinada, a rede neural artificial foi implementada no CLP utilizando a linguagem ladder de programação e posteriormente foi aplicado o sistema de escalonamento de ganhos através da rede neural artificial.

6.1 Normalização das entradas e saídas

Antes que a rede seja embarcada no CLP, segundo Gambogi (2013), é necessário que seja feita a normalização das entradas e saídas.

Como a função de ativação é do tipo tangente hiperbólica sigmoide é necessário que todos os valores de entrada e saída estejam dentro do intervalo de -1 à +1, este cálculo é feito com base nos valores máximo e mínimo das entradas e saídas.

Segundo Gambogi (2013), a normalização pode ser calculada seguindo a equação 4.

.

$$Vn = \frac{(V - Vmin).(Lmax - Lmin)}{Vmax - Vmin} + Lmin \qquad (4)$$

Onde:

Vn : Valor normalizado.

V : Valor anterior a normalização.

Vmax : Valor máximo possível.

Vmin : Valor mínimo possível.

Lmax : Valor do limite máximo possível.

Lmin : Valor do limite mínimo possível.

6.2 Descrição matemática da rede neural e implementação da lógica

O neurônio pode ser escrito agora matematicamente para posteriormente ser programado no CLP, na Figura 9 pode ser visto a rede neural artificial bem como seus componentes.

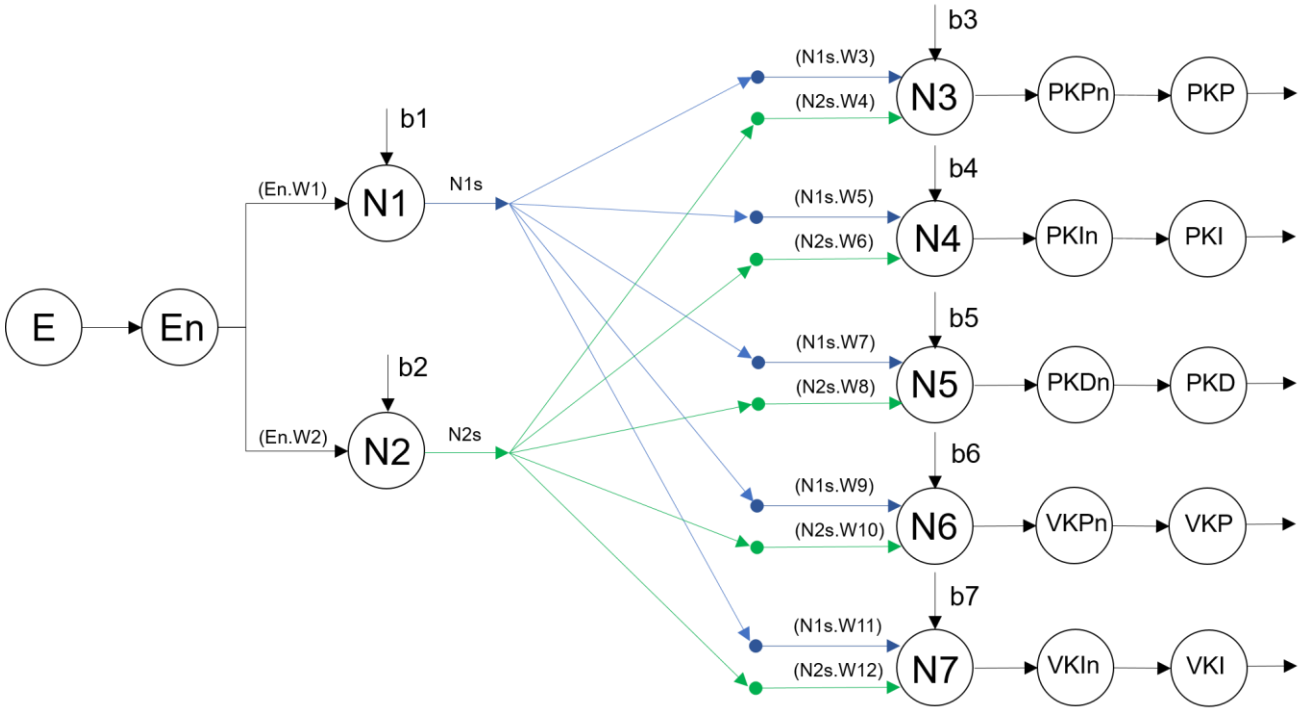

Figura 9 – RNA e seus componentes.

Conforme visto na Figura 8:

E - Representa a entrada do neurônio, ou seja, a posição do servo motor.

En - É a entrada após o processo de normalização conforme visto anteriormente.

N(1-7) - São os neurônios da rede neural artificial.

N(1-7)s - É a saída do neurônio após ter sido feito a somatória e aplicado a função de ativação.

W(1-12) - São os pesos sinápticos.

B(1-7) - São o bias.

PKPn - É o valor do ganho normalizado.

PKP – É o valor do ganho a ser programado no servo *drive*.

Calculando o neurônio com base nos pesos e bias adquiridos no Matlab, uma rotina foi criada no CLP em uma periódica de 2ms e os cálculos foram implementados na linguagem ladder no programa computacional Studio 5000 da Allen Bradley. Todos os pesos sinápticos e bias foram atribuídos a variáveis reais.

Foi feito uma simulação do neurônio para validar se a programação em lógica ladder iria corresponder aos resultados obtidos no Matlab, o resultado foi o esperado, o neurônio programado em ladder obteve a mesma resposta no CLP que obteve no software Matlab para as mesmas entradas.

7. ANALISE DOS RESULTADOS

A planta de selagem foi submetida a testes dinâmicos sem a utilização da rede neural artificial e posteriormente com a utilização da rede neural e o resultado pode ser visto na Figura 10.

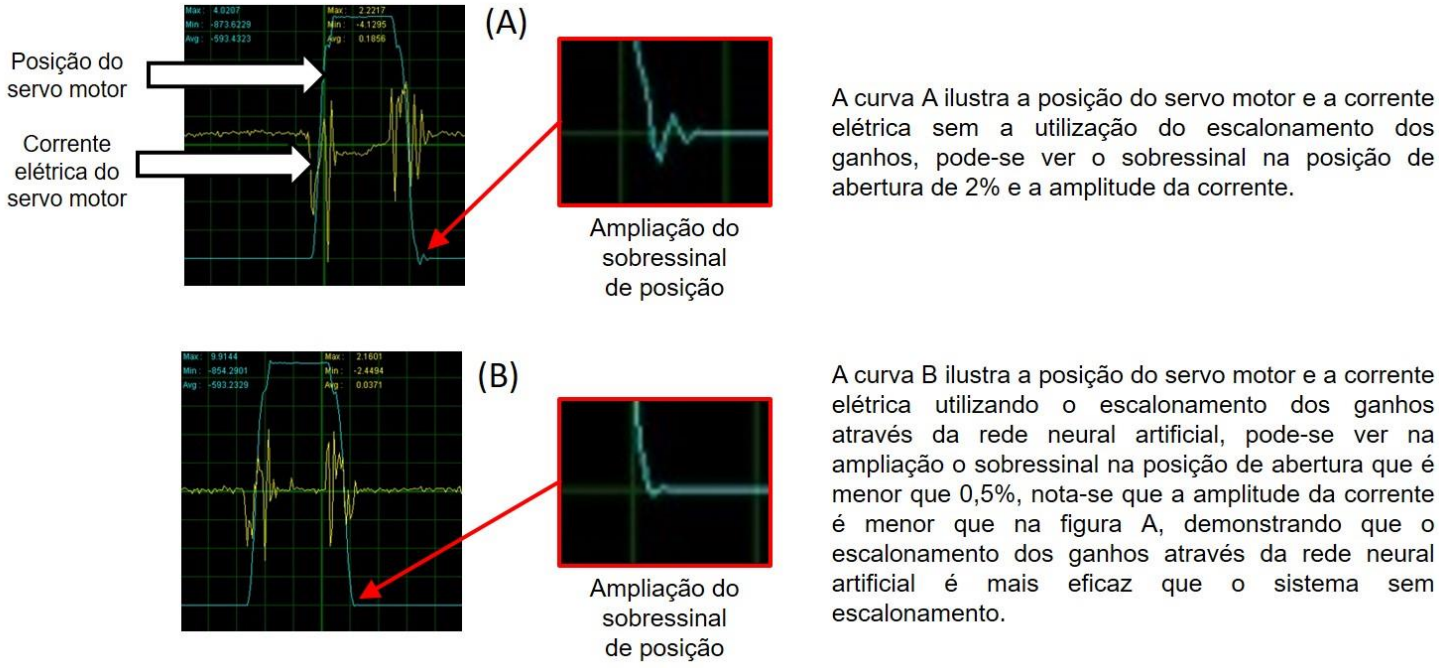

Figura 10 – Comparação dos resultados.

Os ganhos foram ajustados para cada região com sucesso assim como a rede neural que foi simulada no programa computacional Matlab e posteriormente foi embarcada no CLP em linguagem ladder obtendo os mesmos resultados da simulação no Matlab.

A rede neural artificial demonstrou eficácia no escalonamento dos ganhos do controlador, obtendo uma redução do sobressinal de posição para um valor menor que 0,5%.

O escalonamento dos ganhos através da rede neural artificial demonstrou uma amplitude menor de corrente, o que caracteriza como uma vantagem em eficiência energética, a corrente obtida com a utilização da rede neural artificial em comparação com o gráfico obtido sem o escalonamento dos ganhos demonstrou uma redução na amplitude da corrente de cerca de 28%.

8. CONCLUSÃO

O escalonamento de ganhos do controlador PID/PI através de uma rede neural artificial para o mordente de selagem estudado, que possui diferentes faixas de operação, demonstrou ser eficaz para redução do sobressinal de posicionamento de 2% para menor que 0,5% e ainda demonstrou menor consumo de corrente elétrica para o conjunto analisado, demonstrando, através dos resultados, ser uma ferramenta que pode ser implementada no sistema atual e produzir o aperfeiçoamento estabelecido no funcionamento da máquina.

REFERÊNCIAS

ABHISHEK, Kumar et al. Weather forecasting model using Artificial Neural Networks. Procedia technology 4, Bangalore - India, p. 311-318, 2012. Disponível em: <http://www.sciencedirect.com/science/article/pii/S221201731200326X>. Acesso em: 15 de Fevereiro de 2017.

ARAI, Masahiko. Bounds on the number of hidden units in binary-valued three-layer neural networks. Neural Networks, v.6, n.6, p. 855-860, 1993. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0893608005801303>. Acesso em: 03 de Janeiro de 2017.

BEALE, Mark Hudson et al. Neural Network Toolbox: User Guide. V. R2016b, 2016. Disponível em: <https://www.mathworks.com/help/pdf_doc/nnet/nnet_ug.pdf>. Acesso em: 03 de Janeiro de 1017.

BINOT, Daniel Henrique Breda. Configuração de redes neurais artificiais para estimação do volume de árvores. Brazil jornal wood science v05, p. 58-67, 2014. Disponível em: <https://periodicos.ufpel.edu.br/ojs2/index.php/cienciadamadeira/article/view/4067>. Acesso em: 03 de Janeiro de 2017.

FUJITA, Osamu. Statistical estimation of the number of hidden units for feedforward neural networks. Neural Networks, v.11, n.5, p. 851-859, 1998. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0893608098000434>. Acesso em: 03 de Janeiro de 2017.

GAMBOGI, Jarbas Aquiles. Aplicação de redes neurais na tomada de decisão no mercado de ações. Dissertação (Mestre em engenharia elétrica) – Universidade de São Paulo, 2013.

HAYKIN, Simon. Redes neurais: Princípios e práticas. 2.ed. Porto Alegre: Bookman, 2001.

HECHT-NIELSEN, Robert. Kolmogorov's mapping neural network existence theorem. IEEE First international conference on neural networks, California-USA, p. III-11 – III-14, 1987.

HECHT-NIELSEN, Robert. Theory of the backpropagation neural network. International joint Conference, Washington-USA, p. 593-605, 1989.

HIROSE, Yoshio et al. Back-propagation algorithm which varies the number of hidden units. Neural Networks, v.4, n.1, p. 61-66, 1991. Disponível em: <http://www.sciencedirect.com/science/article/pii/089360809190032Z>. Acesso em 03 de Janeiro de 2017.

MIGUEZ, Geraldo Azar. Otimização do algoritimo de backpropagation pelo uso da função de ativação bi-hiperbólica. Tese (Doutor em engenharia de sistemas e computação) – Universidade Federal do Rio de Janeiro, 2012.

SILVA, Ivan Nunes et al. Redes Neurais Artificiais: para engenharia e ciências aplicadas. São Paulo: Artliber, 2010.

MathWorks, Erro médio quadrado. Disponível em: <https://www.mathworks.com/help/images/ref/immse.html>. Acesso em: 03 de Janeiro de 1017.

MathWorks, Regularização Bayesiana. Disponível em: <https://www.mathworks.com/help/nnet/ref/trainbr.html>. Acesso em: 03 de Janeiro de 1017.

MathWorks, Tangente Hiperbólica Sigmoide. Disponível em: <https://www.mathworks.com/help/nnet/ref/tansig.html>. Acesso em: 03 de Janeiro de 1017.