Auto-Compensação de Sensores *Foundation Fieldbus* utilizando Redes Neurais Recorrentes através de Blocos Funcionais

David Ricardo do V. Pereira, Eloi Cagni Júnior, Ricardo Wendell, Adrião D. Dória Neto, Jorge D. Melo
Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte
Natal, Brasil
E-mail: david@dca.ufrn.br, eloi@dca.ufrn.br, wendell@dca.ufrn.br, adriao@dca.ufrn.br, jdmelo@dca.ufrn.br

***Resumo*— O presente trabalho tem como objetivo mostrar um método para realizar a auto-compensação de sensores Fieldbus Foundation utilizando Redes Neurais Recorrentes através de blocos funcionais. As características de tais redes possibilitam o aprendizado da dinâmica da variação da curva de calibração do sensor com o tempo, tornando possível saber o quanto um sensor está descalibrado em um certo momento. Uma introdução sobre a técnica de auto-compensação, de Redes Neurais Recorrentes e do protocolo Fieldbus Foundation será feita e, em seguida, serão mostrados a implementação do algoritmo e os resultados obtidos.**

***Palavras-Chave*— Redes Neurais Recorrentes, Auto-Compensação e Fieldbus Foundation.**

I. Introdução

A Realização de medições precisas é de extrema importância para a indústria do petróleo, tanto por questões de segurança, ao se monitorar a pressão e a temperatura em um oleoduto, por exemplo, quanto por questões de produtividade. Entretanto, independentemente da qualidade de um sensor, ele se desgasta com o tempo, passando a indicar valores que não são coerentes com os valores reais da grandeza física medida. Esse tipo de desgaste recebe nome de descalibração.

A inexatidão na medida faz com que seja necessária uma recalibração no sensor. Para isso, deve-se retirar o sensor do campo, levá-lo ao laboratório e recalibrá-lo. Esse processo pode levar muito tempo e acarretar em prejuízos econômicos, pois a medição deve ser interrompida. Como é desconhecida a periodicidade ideal para se retirar o sensor do campo para realizar o processo de calibração, corre-se o risco de se retirá-lo antes do tempo ideal, o que provocaria uma sub-utilização do sensor, ou ainda fazer a calibração de forma tardia, o que levaria a erros de medição.

A partir dessas informações, foi proposto em [1] o algoritmo da auto-compensação, responsável por corrigir automaticamente os erros decorrentes de uma falta de calibração. A implementação desse algoritmo foi feita utilizando Redes Neurais Perceptron de Múltiplas Camadas treinada pelo algoritmo da retropropagação. Esta rede neural foi utilizada por seu bom desempenho na aproximação de funções e por sua grande capacidade de generalização, i.e. fornecer saídas apropriadas mesmo para entradas que não estiveram presentes durante o processo de treinamento. Os exemplos de treinamento para a rede eram um conjunto de curvas de calibração em diversos momentos do tempo.

O algoritmo, entretanto, precisava ser executado em um ambiente industrial adequado, ou seja, necessitava de uma comunicação em tempo real com os dados provenientes dos sensores. O protocolo que melhor se adaptou foi o Foundation Fieldbus (FF), pois possui uma rede de comunicação rápida, digital e tem suporte para englobar o algoritmo proposto.

O algoritmo foi implementado através de blocos funcionais que possuem funções básicas que, combinados, são capazes de implementar a rede neural necessária para o algoritmo.

Após o treinamento, que era feito de forma *off-line*, os pesos da rede eram inseridos em blocos funcionais específicos do sensor Foundation Fieldbus. O sensor, portanto, iria realizar o processo de *feed-forward* da rede, recebendo como entrada a medida do sensor e um dado informando o seu tempo de funcionamento. Com base nisso a rede conseguia compensar o erro de calibração. Esses dados podem ser vistos na figura 1, que destaca as curvas de calibração no mês 0 e no mês 36.

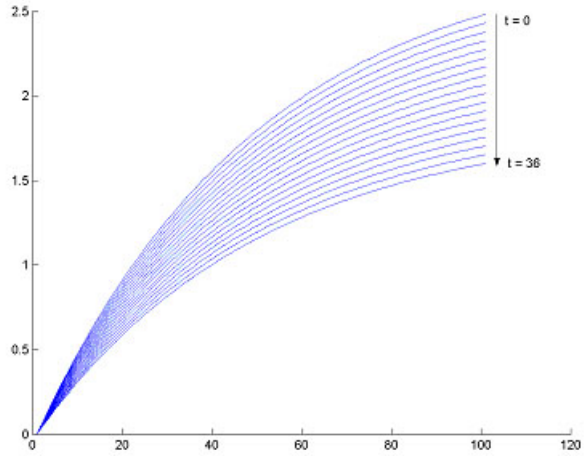

Fig. 1. Gráfico das curvas de calibração do sensor em 36 meses.

Os resultados obtidos foram bastante satisfatórios e o erro médio quadrático final foi bastante baixo, da ordem de 10^{-7} no modo *off-line* em PC, e da ordem de 10^{-1} no modo *on-*

line através do sensor com blocos funcionais. Podemos ver nas figuras 2 e 3 um exemplo de saída *off-line* e *on-line*, onde a linha azul corresponde à curva de calibração do sensor, a curva vermelha é a curva de calibração após dois anos de funcionamento do sensor e a linha preta é a curva vermelha compensada pelo algoritmo da auto-compensação.

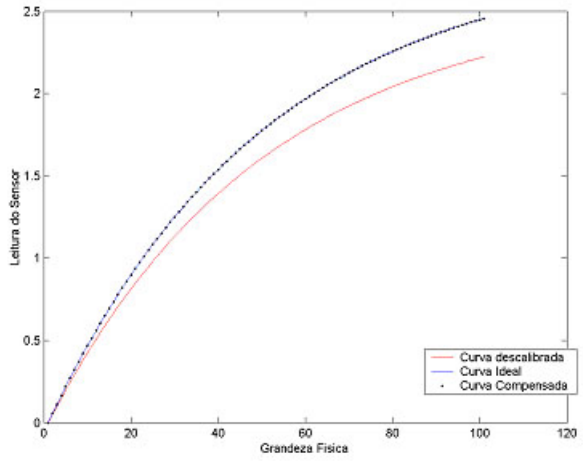

Fig. 2. Resultado da auto-compensação off-line.

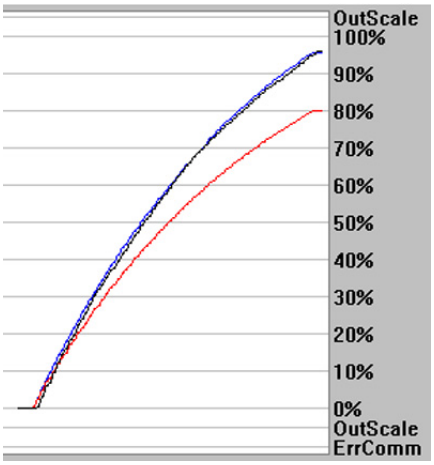

Fig. 3. Resultado da auto-compensação com blocos funcionais.

O problema dessa abordagem é a necessidade de se fornecer ao algoritmo o tempo de funcionamento do sensor Foundation Fieldbus. No ambiente proposto, uma rede Foundation Fieldbus, isso é impossível, o que dificulta a sua utilização. Assim, foi feita uma nova proposta para resolver o problema da auto-compensação: a utilização de Redes Neurais Recorrentes. Essas redes têm a característica de possuem laços de realimentação, ou seja, utilizam como entrada informações de entradas ou saídas passadas.

A dinâmica da variação da curva de resposta do sensor devido à descalibração fornece meios para realizar a compensação da leitura, de modo a obter uma medição próxima de um sensor calibrado. Tal dinâmica é função de diversos fatores que atuam sobre um sensor quando da sua implantação em campo, tais como o tempo decorrido desde a última calibração, a exposição a condições climáticas adversas ou pequenos defeitos. A descalibração em função do tempo resume diversos destes fatores e é um excelente passo inicial para a pesquisa sobre a auto-compensação do sensor. O objetivo deste trabalho é mostrar a nova abordagem para a auto-compensação e compará-la com os resultados obtidos em trabalhos anteriores.

II. Redes Neurais Recorrentes

As redes recorrentes são redes neurais com um ou mais laços de alimentação [3]. A realimentação pode ser de natureza local ou global. Dado um perceptron de múltiplas camadas, a aplicação de realimentação global pode assumir uma variedade de formas. Pode-se ter realimentação dos neurônios de saída ou de neurônios da camada oculta da rede.

A planta arquitetural de uma rede recorrente perceptron de múltiplas camadas assume muitas formas diferentes, contudo elas compartilham as seguintes características comuns:

- Todas elas incorporam um perceptron de múltiplas camadas estático ou parte dele.
- Todas elas exploram a capacidade de mapeamento não-linear do perceptron de múltiplas camadas

A figura 4 mostra a arquitetura de uma rede recorrente genérica que resulta naturalmente de um percetron de múltiplas camadas. O modelo tem uma única entrada que é aplicada a uma memória de linha de atraso derivada com q unidades. Ela tem uma única saída que é realimentada para a entrada de uma outra memória de linha de atraso derivada, também com q unidades. Os conteúdos destas duas memórias de linha de atraso com derivação são utilizados para alimentar a camada de entrada do perceptron. O valor presente da entrada do modelo é representado por $u(n)$, e o valor correspondente da saída do modelo é representado por $y(n+1)$; isto é, a saída está adiantada em relação à entrada por uma unidade de tempo.

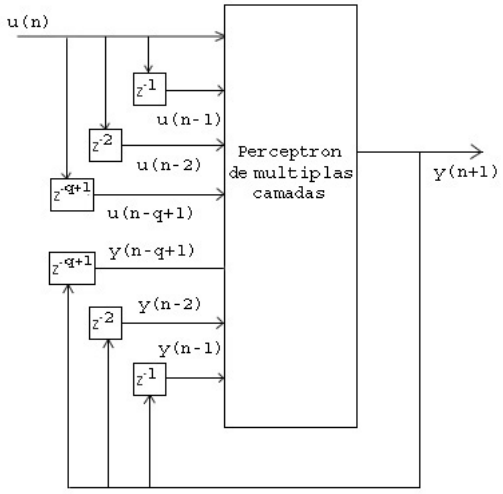

Fig. 4. Modelo auto-regressivo não-linear.

A noção de estado desempenha um papel vital na formulação matemática de um sistema dinâmico. O estado

de um sistema dinâmico é formalmente definido como um conjunto de quantidades que resumem toda a informação sobre o comportamento passado que é necessária para descrever unicamente o seu comportamento futuro, exceto pelos efeitos puramente externos que surgem devido à entrada aplicada. Em termos matemáticos, o comportamento dinâmico do sistema, assumido livre do ruído, é descrito pelo seguinte par de equações não-lineares:

$$\mathbf{x}(n+1) = \varphi(\mathbf{W_a}\mathbf{x}(n) + \mathbf{W_b}\mathbf{u}(n)) \quad (1)$$

$$\mathbf{y}(n) = \mathbf{C}\mathbf{x}(n) \quad (2)$$

onde $x(n)$ é um vetor q-por-1, $u(n)$ é um vetor m-por-1, W_a é uma matriz q-por-q, W_b é uma matriz q-por-$(m+1)$, C é uma matriz p-por-q, e φ é a função de ativação. A dimensionalidade do espaço de estados, ou seja q, é a ordem do sistema.

A matriz W_a representa os pesos sinápticos dos q neurônios na camada oculta que estão conectados aos nós de realimentação na camada de entrada. A matriz W_b representa os pesos sinápticos destes neurônios ocultos que estão conectados aos nós fontes na camada de entrada. Assume-se que os termos de bias para os neurônios ocultos estão incorporados na matriz de pesos W_b A matriz C representa os pesos sinápticos dos p neurônios lineares na camada de saída que estão conectados aos neurônios ocultos. Assume-se que os termos de bias para os neurônios de saída estão incorporados na matriz de pesos C.

A função não-linear $\varphi(.)$ representa a função de ativação sigmóide de um neurônio oculto. A função de ativação tipicamente assume a forma de uma função tangente hiperbólica ou logística.

Uma propriedade importante de uma rede recorrente descrita pelo modelo de espaço de estados é que ela pode aproximar uma ampla classe de sistemas dinâmicos não-lineares. Entretanto, as aproximações são válidas apenas em subconjuntos compactos do espaço de estados e para intervalos finitos de tempo.

Em relação ao treinamento da rede, existem dois modos de se treinar uma rede recorrente:

- Treinamento por época: Para uma dada época, a rede recorrente inicia a execução de algum estado inicial até alcançar um novo estado, em que o treinamento é parado e a rede é reinicializada em um estado inicial para a próxima época. O estado inicial não precisa ser o mesmo para cada época de treinamento. Em vez disso, o que é importante é que o estado inicial da nova época seja diferente do estado alcançado pela rede ao final da época anterior. No treinamento por época, o termo "época"é utilizado em um sentido diferente daquele para um perceptron de múltiplas camadas ordinário. Na terminologia corrente, a época para a rede recorrente corresponde a um padrão de treinamento para o perceptron de múltiplas camadas ordinário.
- Treinamento contínuo: Este método é adequado a situações em que não haja estados de reinício e/ou em que seja requerida aprendizagem em tempo de execução. A característica que distingue o treinamento contínuo é que a rede aprende enquanto realiza o processamento de sinal, ou seja, o processo de treinamento nunca pára.

Baseado nesses modelos, existem dois tipos de algoritmos de aprendizagem para as redes recorrentes:

- O algoritmo da retropropagação através do tempo opera com a premissa de que a operação temporal de uma rede recorrente pode ser desdobrada em um perceptron de múltiplas camadas. A retropropagação através do tempo pode ser implementada no modo por época, no modo contínuo (em tempo real), ou na combinação destes.
- O algoritmo de aprendizagem recorrente em tempo-real é derivado do modelo de espaço de estados.

Estes dois algoritmos compartilham características em comum. Primeiro, ambos são relativamente simples de implementar, mas são lentos pra convergir. Segundo, eles são interrelacionados pois a representação em grafo de fluxo de sinal do algoritmo de retropropagação através do tempo pode ser obtida pela transposição da representação em grafo de fluxo de sinal de uma certa forma do algoritmo de aprendizagem recorrente em tempo-real.

A aprendizagem em tempo-real utiliza a "quantidade mínima de informação disponível", isto é, uma estimativa instantânea do gradiente da função de custo em relação ao vetor de parâmetros a ser ajustado. Pode-se acelerar o processo de aprendizagem explorando a teoria do filtro de Kalman, que utiliza mais efetivamente a informação contida nos dados de treinamento.

III. Protocolo Foundation Fieldbus

De acordo com [2], a *Fieldbus Foundation* [6] é uma organização independente que desenvolve e mantém um padrão de redes de campo para automação de processos, o Foundation Fieldbus.

O Foundation Fieldbus é um sistema de comunicação totalmente digital, que pode ser em série ou bidirecional, conectando equipamentos fieldbus tais como atuadores, sensores e controladores.

O Fieldbus é uma LAN (rede local) para automação e instrumentação de controle de processos, dotado da capacidade de distribuir o controle desses processos no campo. Existem duas redes FF, uma de baixa velocidade concebida para interligação de instrumentos (H1 - 31,25 kbps) e outra de alta velocidade utilizada para integração das demais redes e para a ligação de dispositivos de alta velocidade como CLPs (HSE - 100 Mpbs).

Uma de suas principais características é a organização da camada de aplicação em blocos, que pode ser dividida em três categorias: os blocos de recurso, os blocos transdutores e os blocos funcionais.

Os blocos funcionais são responsáveis por manipular os sinais fornecidos pelos blocos de recurso, depois de tratados pelos blocos transdutores. Eles oferecem interfaces de entrada e saída bem definidas, capacitando-os de se interligarem e formarem novos blocos, o que torna simples a tarefa de melhorar a rede com novas funcionalidades.

Os blocos de funções que serão abordados nesse trabalho, e que são suficientes para a construção da rede neural proposta, serão explicados nas sub-seções seguintes.

A. Bloco Aritmético (ARTH - Arithmetic)

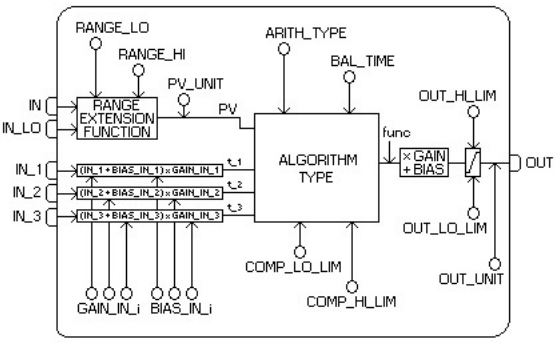

Fig. 5. Modelo do Bloco Aritmético (ARTH).

O bloco aritmético (ARTH) [4] é utilizado para calcular as medidas de combinações dos sinais dos sensores. O bloco possui cinco entradas, onde as duas primeiras são dedicadas a uma função de extensão da rangeabilidade, que resulta em um termo PV, com um status que reflete a entrada em uso. As três entradas restantes são combinadas com esse termo PV para executar uma das funções matemáticas predefinidas pelo bloco, função essa escolhida através do parâmetro *ARITH_TYPE*. Cada uma dessas três entradas adicionais possuem uma constante de polarização (bias) e de ganho. O bias é adicionado e o ganho é multiplicado à soma. O resultado é um valor interno chamado t para cada entrada, de acordo com as equações abaixo:

$$t = (IN + BIAS) * GAIN \tag{3}$$

Após o algoritmo matemático escolhido pelo parâmetro *ARITH_TYPE* ser executado, um bias e um ganho final serão aplicados no resultado.

B. Bloco Caracterizador (CHAR - Characterizer)

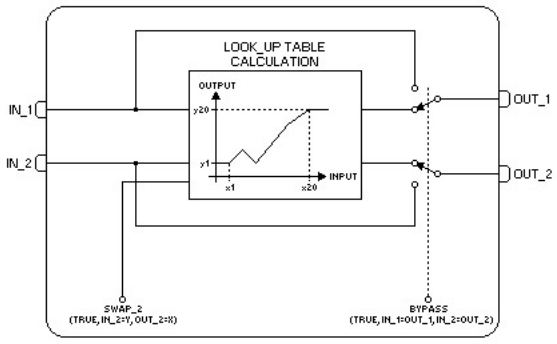

Fig. 6. Modelo do Bloco Caracterizador (CHAR)

O bloco caracterizador (CHAR) [4] tem duas entradas, cada uma com uma saída, que é uma função não-linear das entradas respectivas.

A função é determinada por uma única tabela com coordenadas (x,y) de vinte pontos cada. O status da entrada é copiado para a saída correspondente, assim o bloco pode ser usado no controle ou no sinal do processo.

O bloco interpola e correlaciona a entrada IN_1 à saída OUT_1 e a entrada IN_2 à saída OUT_2 de acordo com uma curva dada pelos pontos:

$$[x_1; y_1], [x_2; y_2] ..... [x_{20}; y_{20}] \tag{4}$$

Se a curva possui *m* pontos, com $m < 20$, então os pontos não configurados, $[x_{m+1}; y_{m+1}]$,$[x_{m+2}; y_{m+2}]$.....$[x_{20}; y_{20}]$ devem ser setados com +INF (mais infinito).

A qualidade e o sub-status de OUT_1 e de OUT_2 refletem o status de IN_1 e de IN_2, respectivamente. Se um dos limites da curva for alcançado, a saída limite se torna constante. Os limites são invertidos se a inclinação da curva for negativa. O status da saída será "Bad - Erro de configuração" se houver um erro, indicado no parâmetro de BLOCK_ERR.

IV. Implementação

As redes neurais recorrentes, por conseguirem captar a dinâmica de um sistema, constituem uma ferramenta poderosa para promover a compensação da leitura do sensor.

Para fins de experimentação do algoritmo, foi definida uma curva hipotética para um sensor e uma dinâmica de amortização de sua leitura em função do tempo (em meses), de acordo com a equação 5.

$$C(t) = (2.9 - 0.03t) * (1 - \exp^{(-0.2x)}) \tag{5}$$

onde t é o tempo do sensor e x a medida do sensor.

A arquitetura da rede foi definida de acordo com as seguintes características desejadas:

- A entrada da rede consiste do valor medido pelo sensor em um dado momento. Foi estabelecido que será tomada uma medição por dia para fins de estabelecer a compensação adequada para o sinal.
- Serão realimentados os valores dos pares entrada-saída de 30 e 60 dias anteriores, de modo a estabelecer a dinâmica da descalibração do sensor.

O mapeamento entrada-saída definido para a rede pode ser visto na figura 7.

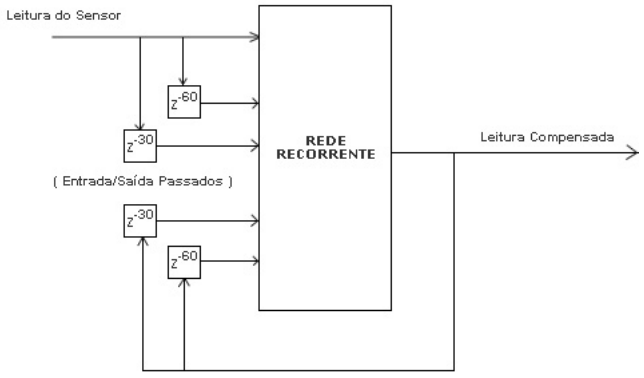

Fig. 7. Modelo da rede recorrente implementada.

A rede neural recorrente foi treinada utilizando o algoritmo da retropropagação no tempo. Após um teste com várias arquiteturas foi selecionada a arquitetura 5x5x1 para a rede, como aquele que obteve um bom resultado.

A implementação utilizando blocos funcionais padrões previstos pela Norma Foundation Fieldbus se baseou no simulador feed-forward, já com os ganhos sinápticos e bias adequados, obtidos no PC de forma *off-line*.

A programação utilizando blocos funcionais é ”visual”, ou seja, o usuário não digita o código do algoritmo que se quer implementar mas sim manipula blocos, modifica certos parâmetros e realiza comunicações (ligações) entre eles.

Para a construção de um neurônio artificial, de acordo com [7], é preciso implementar com os blocos as seguintes funções:

$$u_k = \sum_{j=1}^{m} w_{kj} * x_j \tag{6}$$

$$y_k = \varphi\left(u_k + b_k\right) \tag{7}$$

onde a saída y_k é obtida através da passagem por uma função de ativação (tangente sigmóide) do somatório do produto do ganho pela entrada, somado com um bias.

De acordo com a figura 5, o bloco aritmético possui três entradas que podem ser somadas por um bias individual e multiplicadas por um ganho individual. Após isso, escolhe-se um dos algoritmos matemáticos predefinidos, através do parâmetro *ARITH_TYPE*. De acordo com a equação 6, o algoritmo necessita executar somente um somatório, portanto seta-se o parâmetro *ARITH_TYPE* para *traditional summer* (somador tradicional). Com isso, os resultados dos três produtos são somados, e nesse resultado será aplicado novamente um bias e um ganho final.

Como um bloco aritmético só possui 3 entradas válidas para a construção do neurônio artificial e a arquitetura da rede recorrente implementada (5x5x1) necessita de 5 entradas para cada neurônio, então utiliza-se uma junção de 3 blocos aritméticos e mais o caracterizador para construir o neurônio artificial, aumentando a quantidade de entradas úteis para 6. Nesse neurônio, os dois primeiros blocos realizam a soma dos produtos, e o terceiro realiza a soma das saídas desses dois blocos, adicionados pelo bias final.

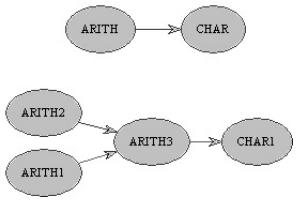

Fig. 8. Neurônios construídos com blocos funcionais para 3 e 6 entradas.

Agora a saída desse bloco aritmético (y) terá de passar por uma função de ativação, uma tangente sigmóide para os neurônios ocultos. Um bloco que pode construir uma função desse tipo é o bloco caracterizador, figura 6. Este bloco pode ter até 20 pontos (x,y), onde esses pontos são interpolados, formando a curva desejada. A entrada IN funciona como um ponto do eixo x, e a saída será o seu respectivo ponto no eixo y, de acordo com a curva.

Como esse bloco caracterizador oferece somente 20 pontos, a curva modelada se torna imprecisa, podendo a margem de erro se manter fora do aceitável. Uma solução seria escolher pontos mais representativos da curva tangente sigmóide para aumentar a precisão, ou então, formar a curva com dois ou mais blocos aritméticos, fazendo com que o número de pontos disponíveis seja bem maior. Um exemplo de curva tangente sigmóide construída com dois blocos (até 40 pontos) pode ser vista abaixo:

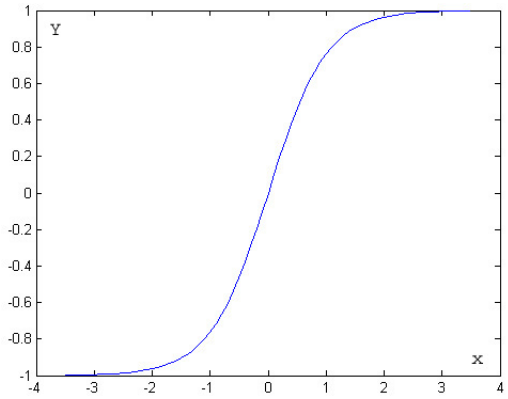

Fig. 9. Curva tangente sigmóide obtida com dois blocos CHAR.

Para esta arquitetura (5x5x1) da rede recorrente, a opção de implementação com blocos funcionais padrões escolhida foi: neurônios com 6 entradas, passando por funções de ativação representadas por curvas formadas pelos 20 pontos mais representativos. Esta arquitetura pode ser vista na figura 10.

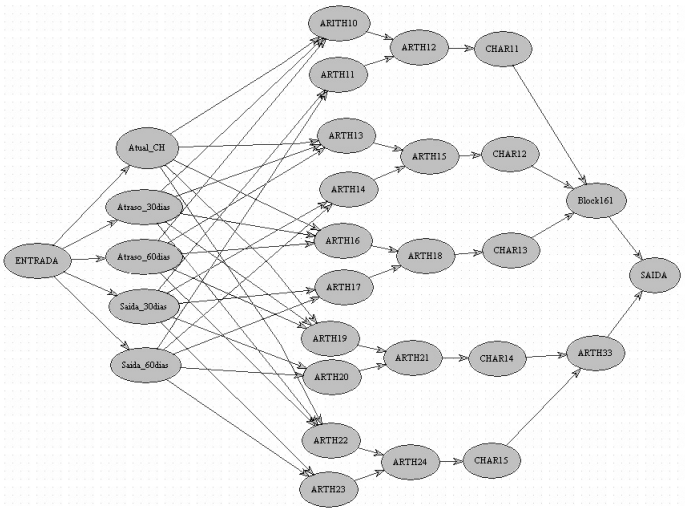

Fig. 10. Rede recorrente implementada com blocos funcionais padrões.

As cinco entradas da rede foram representadas por blocos caracterizadores que representavam as curvas de calibração nos instantes de tempo, sendo inicializados por um bloco funcional cuja saída é uma função rampa.

V. Resultados

Os resultados obtidos pelo método de auto-compensação proposto podem ser vistos nas figuras abaixo. Essas figuras

mostram os resultados obtidos com 10 e 30 meses de funcionamento do sensor no modo *off-line* e no modo *on-line* (com o TagView da Smar [5]), respectivamente. A linha azul corresponde à curva de calibração ideal do sensor, a linha verde ou vermelha é a curva de calibração degradada (10 e 30 meses) e a linha preta correspondem à curva do sensor compensada.

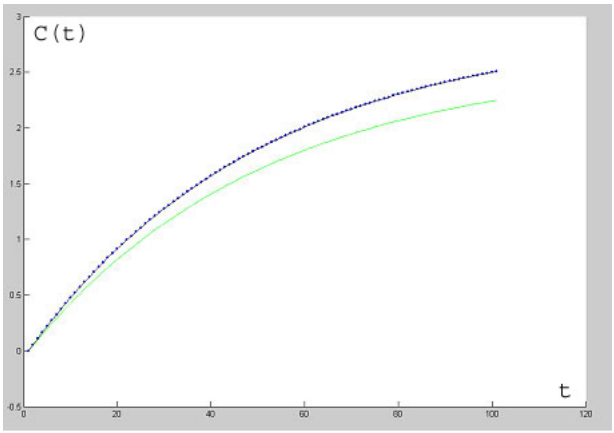

Fig. 11. Resultado *off-line* para 10 meses.

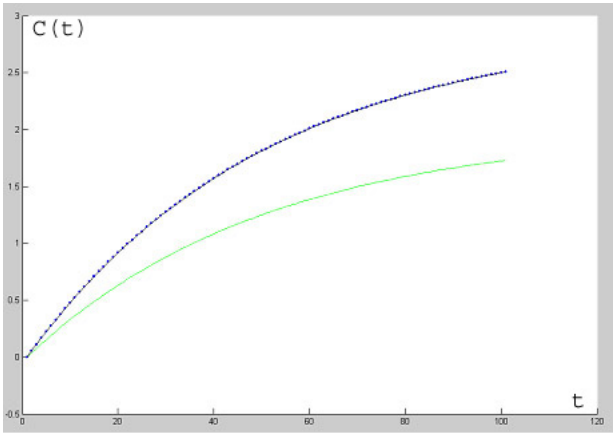

Fig. 12. Resultado *off-line* para 30 meses.

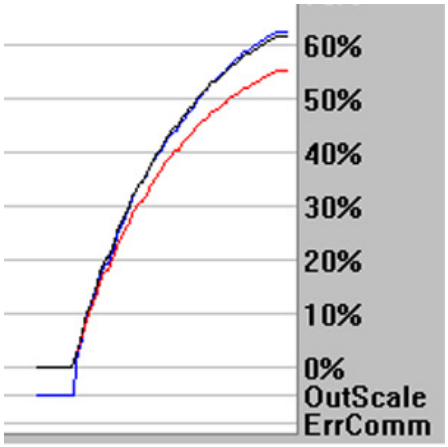

Fig. 13. Resultado *on-line* para 10 meses.

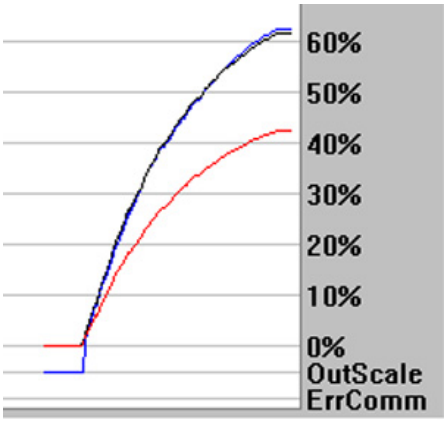

Fig. 14. Resultado *on-line* para 30 meses.

Os resultados obtidos com o treinamento da rede foram bastante satisfatórios e o erro médio quadrático foi da ordem de 10^{-7} no modo *off-line* e 10^{-1} no modo *on-line*. Pode-se perceber que os resultados foram bastante semelhantes aos obtidos com o método utilizado em [1], sendo a diferença de precisão entre eles desprezível. Visto o bom funcionamento da Auto-Compensação, pode-se deduzir que a Auto-Calibração, que obedece o mesmo mecanismo de acompanhamento de curva, também irá ter um bom funcionamento na implementação com blocos. E a Auto-Validação terá um bom funcionamento se os dois outros algoritmos também o tiverem no ambiente de trabalho do padrão Fieldbus.

VI. Conclusão

Tendo em vista a sua grande necessidade de exatidão nas medições, a indústria do petróleo tem feito muitos investimentos em pesquisas com sensores e o algoritmo da auto-calibração pode melhorar bastante o tempo de uso do sensor. As Redes Neurais Recorrentes possibilitam o uso do algoritmo de um modo mais generalizado, já que com o seu uso o algoritmo não necessita do tempo de funcionamento do sensor como entrada. Os resultados obtidos com os experimentos foram satisfatórios e estão de acordo com o esperado.

Referências

[1] Pereira, D. R. V., Bezerra, J. P. M., Dória Neto, A. D. e Melo, J. D. *Self-Compensation, Self-Calibration and Self-Validation of Foundation Fieldbus Intelligent Sensors*. VI Induscon - Conferência Internacional de Aplicações Industriais. Joinville-SC, 2004.

[2] Coretti, Eng. José Alberto, Manual de Treinamento - Foundation Fieldbus, Smar, 2002.

[3] Haykin, Simon, *Redes Neurais: Princípios e Prática*, Bookman, 2001.

[4] Fieldbus Foundation. Function Block Application Process. In Foundation^{TM} Specification, 2001.

[5] http://www.smar.com/, acesso em 28/04/2005

[6] http://www.fieldbus.org/, acesso em 28/04/2005

[7] Silva, Diego R. Cabral. Tese de Mestrado. *Redes Neurais Artificiais no Ambiente de Redes Industriais Foundation Fieldbus Usando Blocos Funcionais Padrões*. Natal-RN, 2005

[8] Cagni Jr, Eloi. *The Implementation of the Self-calibration, Self-compensation and Self-validation Algorithms for Foundation Fieldbus Sensors are presented using Standard Function Blocks*. CIMSA 2005 - 2005 IEEE International Conference on Computational Intelligence for Measurent Systems and Applications. Taormina, Sicily, Italy.

[9] Silva, Diego R. Cabral, Lima, Fabio. *Implementação de Redes Neurais Artificiais em Ambientes de Redes Industriais Foundation Fieldbus com o uso de Blocos Funcionais Padrões*. CBRN - Congresso Brasileiro de Redes Neurais. Natal-RN, 2005.