Redes Neurais para Controle de Atitude de Satélites

Valdemir Carrara[1], Atair Rios Neto[2]
[1]Instituto Nacional de Pesquisas Espaciais – INPE/MCT
CEP 12201-970 CP 515 São José dos Campos, SP - Brazil
[2]Instituto de Pesquisa e Desenvolvimento, Universidade do Vale do Paraíba.
CEP 12245-720 São José dos Campos, SP – Brazil
E-mails: val@dem.inpe.br, atair@univap.br

Abstract

Feedforward neural networks are investigated in this work to verify their ability to control the attitude of a satellite. Neural nets are a promising tool for attitude control due to its inherent nonlinear behavior, which makes them a natural candidate to control nonlinear systems. Nevertheless, as will be shown, to obtain a neural 3 axis attitude controller is not as simple as a conventional SISO net control. Main difficulties are the large amount of training data, in order to assure the complete understanding of the attitude dynamics by the neural net, and also the fact that attitude control is a MIMO instead of a SISO system. Regardless of these draw backs, some results concerning attitude control will be shown.

1. Introdução

Redes neurais (RN) têm sido empregadas para controle de sistemas não-lineares, com base na premissa de que poderiam desempenhar esse papel por serem aproximadores universais de funções não-lineares limitadas. Aliada a esta característica, as RNs teriam ainda a vantagem do aprendizado, que dispensa totalmente ou em parte a necessidade de modelagem matemática do sistema. Diversos métodos diferentes foram propostos para tal finalidade [1] e [2], já que o desempenho de cada um deles depende das peculiaridades do sistema a ser controlado. No entanto, como será mostrado, a obtenção de uma RN para controlar a atitude de um satélite artificial é dificultada pelo fato de tratar-se de um sistema MIMO ('multiple input, multiple output'). Em contrapartida, a grande maioria dos sistemas controlados através de RNs são sistemas SISO ('single input, single output') [2] e [3]. A complicação deve-se à grande quantidade de pontos de treinamento, de forma a assegurar um mapeamento completo da dinâmica de atitude pela rede neural, agravada pelo fato de se ter um sistema com múltiplas entradas. Para que a rede adquira as características do sistema, é necessário que ela aprenda o comportamento do sistema inverso, isto é, dado o estado desejado para planta, a rede deve obter o sinal de controle que levará o sistema a este ponto. A dinâmica, neste caso, deve ser tal que admita a existência de somente uma solução inversa na trajetória de referência, para que o aprendizado seja possível. Nem sempre, infelizmente, esta premissa é verdadeira, e com isso o treinamento não converge.

Diversos métodos foram propostos no sentido de obter a dinâmica inversa e garantir a convergência. Entre esses, os métodos generalizado, especializado e o controle preditivo ([1] e [2]) adaptam-se melhor a certos tipos de problemas. Em geral, usam uma trajetória de referência calculada previamente ou calculada com base no estado atual do sistema para alimentar uma RN do tipo 'feedforward' atuando como controlador. Entretanto, em certas aplicações, é mais importante corrigir erros residuais do estado do que simplesmente seguir uma trajetória em malha aberta. O controle de atitude de satélites é um bom exemplo deste último, no qual pequenas porém significativas perturbações desviam a orientação do aparelho com o tempo, fazendo com que as antenas ou sistemas óticos deixem de apontar para a direção certa. Aliado a isso, a dinâmica da atitude é não-linear, o que torna a correção e controle da atitude um processo complicado. Caso seja utilizado uma trajetória de referência estática, esta não terá como prover a rede controladora com a informação dinâmica necessária para compensar possíveis desvios da atitude. Nesses sistemas, onde um controle em malha fechada é essencial, sugere-se uma rede alimentada com uma trajetória obtida a partir do erro no estado. O sinal de erro tem a característica de produzir um controle nulo quando o próprio erro for nulo, e traz consigo informações suficientes para permitir ajustes no estado do sistema. Por sua vez, a rede neural pode aprender o comportamento não-linear do satélite, gerando um controle não-linear baseado na realimentação do erro. Esse método é semelhante ao generalizado inverso [1], com a diferença que o sistema, agora, é controlado em malha fechada.

Para viabilizar e testar o método, o comportamento dinâmico de um satélite foi simulado em ambiente computacional [4], já que é impraticável realizar o treinamento da rede com o satélite já em órbita. Foi utilizada uma rede 'feedforward' de duas camadas, com função de ativação sigmóide na primeira e com função linear na segunda. O número de neurônios na rede e o número de pontos de treinamento foram ajustados

interativamente de forma a se obter o menor tempo de treinamento possível. Os resultados da simulação foram comparados com um controlador PD (proporcional e derivativo) convencional. Verificou-se que o PD tem desempenho melhor do que a rede neural, no que se refere exclusivamente ao tempo de resposta. As redes neurais têm, no entanto, potencial para reverter a desvantagem, caso utilize-se de suas propriedades como o processamento paralelo e o tratamento simultâneo de vários elementos sensoriais.

2. Redes neurais

Redes neurais artificiais são elementos compostos de unidades individuais denominadas neurônios, agrupados em camadas. Em redes do tipo 'feedforward' cada neurônio aplica uma função de ativação f ao somatório das entradas ponderadas vindas da camada anterior. As primeiras camadas possuem em geral uma função de ativação não-linear limitada como por exemplo a função sigmóide ou a tangente hiperbólica. A última camada consiste basicamente de um agrupador linear das saídas das camadas precedentes [1]. Os pesos utilizados nas ponderações das entradas são obtidos através de métodos de otimização, denominados treinamento ou aprendizado, que minimizam o erro apresentado pela rede a cada iteração. Uma rede de duas camadas, com função de ativação do tipo sigmóide na camada de entrada:

$$f(x) = \frac{1 - e^{-x}}{1 + e^{x}} \tag{1}$$

e linear na última, pode aproximar com uma dada precisão qualquer função não-linear limitada [5] e [6]. Uma rede 'feedforward' composta por l camadas, como mostrado na Figura 1, pode ser vista como uma função que realiza um mapeamento, associando vetores compostos por n_0 elementos de entradas com vetores de n_l componentes na saída. Representando o peso associado à j-ésima entrada do i-ésimo neurônio da camada k (provinda do j-ésimo neurônio da camada precedente) por w_{ij}^k e a função de ativação desta camada por f^k, então este apresenta em sua saída o valor x_i^k dado por:

$$x_i^k = f^k\left(\bar{x}_i^k + b_i^k\right) = f^k\left(\sum_{j=1}^{n_{k-1}} w_{ij}^k \, x_j^{k-1} + b_i^k\right) \tag{2}$$

onde b_i^k é o patamar de ativação do neurônio, ou 'bias', que permite o neurônio responder com uma saída não nula mesmo quando as entradas consistirem de um vetor nulo.

Geralmente o patamar é obtido junto com o processo de determinação dos pesos, através da adição de um novo componente no vetor de entradas, com valor unitário. Numa forma matricial, considerando o vetor de saídas da camada k como x^k, tem-se:

$$x^k = f^k\left(\bar{x}^k\right) = f^k\left(W^k x^{k-1}\right) \tag{3}$$

sendo que a matriz de pesos W^k, inclui os patamares:

$$W^k = \begin{bmatrix} w_{11}^k & \cdots & w_{1n_{k-1}}^k & b_1^k \\ w_{21}^k & \cdots & w_{2n_{k-1}}^k & b_2^k \\ \vdots & & \vdots & \vdots \\ w_{n_k 1}^k & \cdots & w_{n_k n_{k-1}}^k & b_{n_k 1}^k \end{bmatrix} \tag{4}$$

cuja dimensão é dada por $n_k \times n_{k-1}+1$. Quanto maior o número de camadas, melhor a representação da rede, isto é, menor é o erro ao final do processo de treinamento [7] e [8], mesmo considerando o mesmo total de neurônios. Por outro lado, a capacidade de generalização, ou seja, a habilidade da rede responder com um erro pequeno em situações na qual não houve treinamento é em geral melhor em redes com poucas ou apenas uma camada oculta [9]. O número de neurônios nas camadas ocultas é importante do ponto de vista do grau de aproximação efetuado pela rede: poucos neurônios leva a um erro elevado na saída, enquanto que muitos neurônios provoca uma oscilação entre os pontos treinados [10].

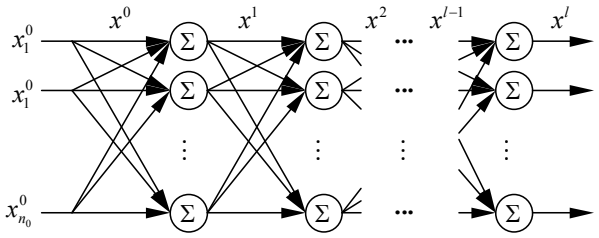

Figura 1: Uma rede neural do tipo 'feedforward '.

3. O método de retropropagação

As maiores vantagens das redes neurais, quando comparadas com outros métodos de aproximação, consistem na sua estrutura paralela, na capacidade de lidar corretamente com um grande número de elementos de entrada, e no aprendizado. A estrutura paralela permite a utilização de processadores paralelos, diminuindo assim o tempo de resposta e do treinamento. A capacidade de processar várias entradas habilita-as a extraírem a correlação adequada entre estas e a saída. Os métodos de treinamento empregados no ajuste dos pesos permitem que a RN adquira a representação do sistema com ajuste empírico de alguns parâmetros. Em geral estes métodos utilizam um critério de otimização com o intuito de minimizar uma função de desempenho associada ao erro ε apresentado pela rede. Eles dependem, portanto, do conhecimento de como o erro é afetado por uma mudança nos pesos. A técnica de

retropropagação do erro [1] obtém a derivada parcial dos elementos do vetor de saída com relação ao *j*-ésimo peso associado ao *i*-ésimo neurônio da camada *k*:

$$\frac{\partial x^l}{\partial w_{ij}^k} = \Delta^k \begin{bmatrix} 0 \\ \vdots \\ x_j^{k-1} \\ \vdots \\ 0 \end{bmatrix}, \qquad (5)$$

onde Δ^k é a matriz de retropropagação do erro, obtida através da relação recorrente:

$$\Delta^k = \Delta^{k+1} W^{k+1} F^k \qquad (6)$$

com condição inicial dada por $\Delta^l = F^l$, sendo que F^k é uma matriz diagonal quadrada com as derivadas da função de ativação f^k em relação ao seu argumento:

$$F^k = \frac{df^k(\bar{x}^k)}{d\bar{x}^k} = \begin{bmatrix} f^{k\prime}(\bar{x}_1^k) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & f^{k\prime}(\bar{x}_{n_k}^k) \end{bmatrix} \qquad (7)$$

Atualmente o método do gradiente é o mais empregado no treinamento de redes neurais. Ele é bastante simples de ser implementado em computadores, é bastante rápido nos cálculos mas apresenta uma convergência muito demorada. De fato, esta é a razão dos tempos de treinamento extremamente elevados na maioria das aplicações de RN. A atualização dos pesos pelo método do gradiente, a cada nova interação, é realizada na forma:

$$W^k(t+1) = W^k(\mathrm{t}) + \lambda\, \Delta^{k^T} \varepsilon\, x^{k-1^T} \qquad (8)$$

sendo λ o parâmetro de convergência do aprendizado, compreendido entre 0 e 1. Embora existam outros métodos que convergem mais rapidamente, como o de mínimos quadrados [8], filtro de Kalman [11], ou o de Levenberg-Marquardt [12] e [13], a simplicidade e a pouca necessidade de memória do método do gradiente asseguram a continuidade do seu uso em situações específicas. Independentemente do método de treinamento, os pesos são atualizados a cada apresentação do vetor de entradas, no denominado aprendizado adaptativo, ou ao final da apresentação de um conjunto completo dos vetores de entrada, conhecido como treinamento por lote.

O aprendizado adaptativo permite o treino em tempo real, com o sistema físico provendo as informações para a rede. O inconveniente do aprendizado adaptativo é que, se o sistema possuir uma dinâmica muito lenta, os pesos serão ajustados em cada ponto de operação, e com isso a rede tem dificuldade em adquirir um conhecimento distribuido e completo acerca do sistema. O treinamento também pode ficar incompleto caso a rede não tenha informações dos vetores de entrada em todo o espaço de fases. O treinamento em lote parece contornar o primeiro problema, mas ainda assim nem sempre pode-se garantir que todo o espaço de fases será percorrido. No caso do controlador de atitude, o treino não pode ser realizado em órbita, após o lançamento do satélite, pois a geração de sinais de controle durante a fase de treinamento, enquanto a rede ainda não fornece valores corretos, poderá colocá-lo numa situação de perigo. Logo, deve-se utilizar a simulação computacional para garantir a funcionalidade do controlador antes de submetê-lo a uma situação real. Este procedimento também permite que o treino seja realizado em todo o espaço de fase e não apenas numa trajetória particular.

Controlar um satélite com redes neurais apresenta também outros problemas, entre os quais encontra-se o grande número de variáveis de estado (ou graus de liberdade do sistema), o que faz com que sejam necessários muitos pontos para que a rede aprenda o comportamento. Considere-se, por exemplo, um sistema com 6 graus de liberdade, como um satélite (3 variáveis para a posição angular e 3 para a velocidade angular), e mais 3 variáveis para o controle. Supondo que 5 amostras sejam suficientes para cada variável, a combinação delas leva a quase 2 milhões de possibilidades nos quais a rede deve ser treinada, praticamente inviabilizando o aprendizado considerando a velocidade e memória dos computadores atuais. Nessas condições o processo de treinamento poder durar vários meses, e a rede resultaria tão grande que seria muito difícil a aplicá-la em tempo-real. Felizmente, pelo menos em teoria, o conjunto de pontos não precisa ser tão grande, pois a rede pode generalizar e adquirir informações suficientes acerca da dinâmica interpolando adequadamente os valores de entrada. Sob esse ponto de vista um método de exploração estatística do espaço de fase é mais conveniente do que a seleção de pontos igualmente espaçados nesse espaço. O número de pontos a ser considerado no treinamento, bem como o número de neurônios da rede irão depender do grau de aproximação desejado para o controle.

4. Métodos de treinamento

Diversos métodos foram estabelecidos [1] na tentativa de obter uma rede neural como elemento controlador de um sistema. Os principais, entre eles o método generalizado inverso, o método indireto e o especializado inverso, apresentam características que os distinguem quanto à sua adaptação ao sistema físico. O problema principal reside no fato de que a rede deve assumir o comportamento inverso do sistema, ou seja, dada a trajetória, qual é o sinal do controle que leva o sistema a seguí-la. Nem sempre o sistema apresenta uma relação direta entre o sinal de controle e o seu estado, residindo aí a maior dificuldade no treinamento em

tempo-real. No treinamento generalizado inverso e também no método indireto, não se pode garantir que o treinamento seja efetuado em todo o espaço de fases, e, assim, instabilidades podem surgir quando a rede for utilizada para controlar o sistema. O método especializado inverso requer uma rede neural funcionando como um modelo do sistema, para que possa ser estabelecida uma correspondência entre o erro na saída do modelo direto e o erro na RN de controle. Se o sistema for simulado computacionalmente, os métodos generalizado e especializado tornam-se equivalentes, e, desde que é sempre possível cobrir todo o espaço de fase numa simulação de atitude, adotou-se o modelo generalizado inverso, mostrado na Figura 2, no treinamento da rede.

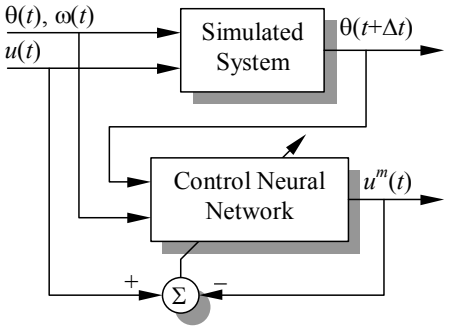

Figura 2: Método generalizado inverso.

As entradas para a rede de controle são os 3 ângulos de erro de atitude $\theta(t)$ e as componentes da velocidade angular $\omega(t)$, no instante t, além do desvio propagado (integração numérica) $\theta(t+\Delta t)$. A saída da rede é composta pelo sinal de controle, ou seja, pelo torque externo aplicado ao satélite, u(t). A atitude foi simulada considerando um satélite rígido não perturbado, com inércias de 23, 23 e 11 kgm^2. Supõe-se que o torque seja provido por jatos de gás (hidrazina), regulado por um sistema de PWM. O torque máximo disponível é de u_{max} = 1.5 Nm. Utilizou-se uma rede tipo 'feedforward' com duas camadas, com funções de ativação sigmóide na primeira e linear na segunda. A rede foi treinada utilizando-se o algoritmo de Levenberg-Marquardt [12] e [13].

O erro de treinamento depende do número de neurônios presentes na primeira camada, e também do número de pares de entrada-saída fornecidos à rede ou pontos de treinamento. Para um dado erro de treinamento, quanto mais pontos de treinamento mais neurônios serão necessários. Pode-se associar esse erro com uma resolução mínima do torque máximo, da ordem de 1%, por exemplo. É claro que quanto menor esse erro tanto maiores serão o número de neurônios e o tempo de treinamento. Infelizmente, ainda não existe uma teoria que forneça o número mínimo de neurônios em função do erro ao término do treinamento. Adotou-se assim um procedimento iterativo: o treinamento é realizado inicialmente numa rede com poucos neurônios e poucos pontos de treinamento, gerados aleatoriamente. Se a precisão não for atingida, então o número de neurônios é incrementado por um fator maior do que 1 (foi utilizado 1.4). Por outro lado, se o treinamento for bem sucedido, então geram-se novos valores de entrada e calcula-se qual o erro médio apresentado pela rede nesses pontos. Se esse erro for ainda inferior ao estipulado para o treinamento, então o treinamento está completo. Caso contrário o número de pontos de treinamento deve ser aumentado por um fator maior do que 1 (1.4, no caso), e todo o processo é repetido. Com esse algoritmo o número de neurônios e o número de pontos crescem até atingir o mínimo necessário para que a rede aprenda a dinâmica do sistema.

5. Resultados de simulação

No problema proposto, o treinamento foi realizado gerando-se posições aleatórias da atitude, compreendidas entre ± 20° e ± 1 rpm em cada um dos 3 eixos. Iniciou-se o procedimento descrito acima com 8 neurônios na primeira camada e 256 pontos de treinamento. O treino prosseguiu e finalmente atingiu a precisão exigida com 24 neurônios e 2576 pontos, conforme mostra a Figura 3.

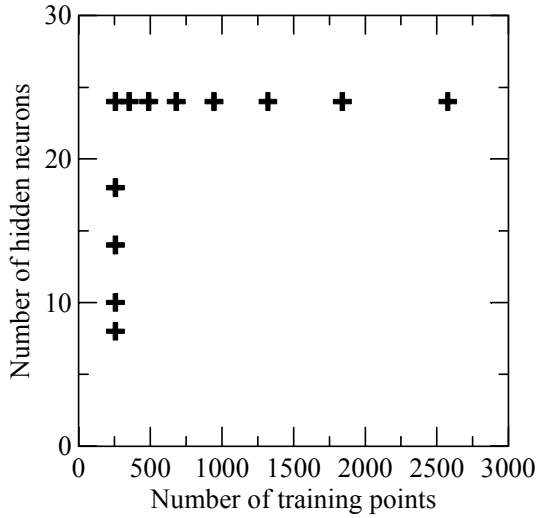

Figura 3: Evolução do treinamento da RN de controle.

Uma vez treinada, a RN de controle foi utilizada junto com um simulador de atitude de forma a validar o controlador. Mas, como mostrado na Figura 2, a rede necessita como um dos parâmetros de entrada a atitude no instante $t+\Delta t$. Esta informação, não disponível no instante presente, deve ser substituída pela trajetória de referência $\theta^r(t+\Delta t)$, como sugerido em [1] e mostrado na Figura 4. A desvantagem deste método é que, infelizmente, ele leva o sistema a se comportar em malha aberta e, portanto, ele não corrige o erro na saída da planta. Foi adotado, então, uma trajetória de referência calculada com base no erro de atitude apresentado na saída da planta, extrapolado para o instante seguinte, de forma a fechar a malha de controle.

Utilizou-se inicialmente uma trajetória de referência proporcional ao erro calculada por:

$$\theta^r(t+\Delta t) = \theta(t) - a_p\,[\theta(t) - \theta^t(t)] \qquad (8)$$

sendo $\theta^t(t)$ a atitude desejada no instante t. Esta trajetória levou o sistema para a atitude correta, como esperado, mas com uma oscilação crescente, pois o satélite não consegue amortecer os movimentos angulares induzidos pelo sinal de controle. Foi necessário, portanto, incluir um fator de amortecimento no cálculo da trajetória de referência, proporcional à velocidade angular, resultando:

$$\theta^r(t+\Delta t) = \theta(t) - a_p\ \theta(t) + a_d\ \omega(t) \qquad (9)$$

já que a atitude desejada é nula.

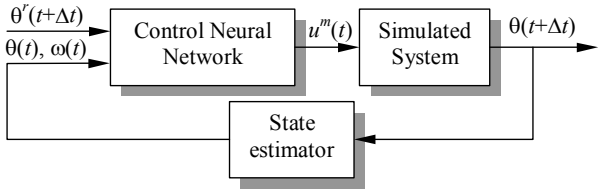

Figura 4: RN atuando como controlador de um sistema.

Mesmo assim, a estabilidade do controle foi bastante difícil de ser conseguida, pois os escalares a_p e a_d tiveram que ser ajustados por tentativa e erro. Pequenas diferenças nos valores de a_p e a_d levaram o sistema a apresentar um tempo de acomodação elevado ou mesmo instabilidade. O melhor resultado foi conseguido com a_p = 0.08 e a_d = 1.2, mostrado numa simulação de 200 segundos nas Figuras 5, 6 e 7.

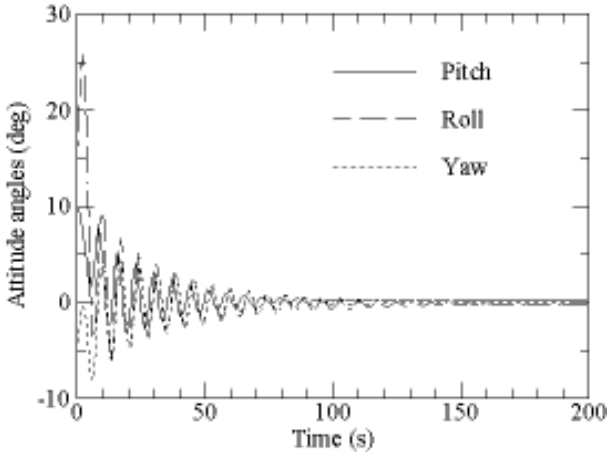

Figura 5: Simulação da atitude com a RN controladora

Como visto nas figuras, a oscilação nos eixos do satélite ('pitch', 'roll' e 'yaw') não foi amortecida totalmente em 200 segundos. A RN controladora apresentou ainda, ao término do treinamento, um patamar de ativação pequeno porém não nulo, o que fez com que o controle respondesse mesmo depois que o sistema tivesse atingido a atitude desejada. Isto provocou um amortecimento muito lento nesta região, e causou um desvio constante na atitude. As condições iniciais adotadas nas Figuras 5 e 6 foram: atitude θ = (10°, 15°, -5°) e velocidade angular ω = (0.1, 0.6, 0.2) rpm.

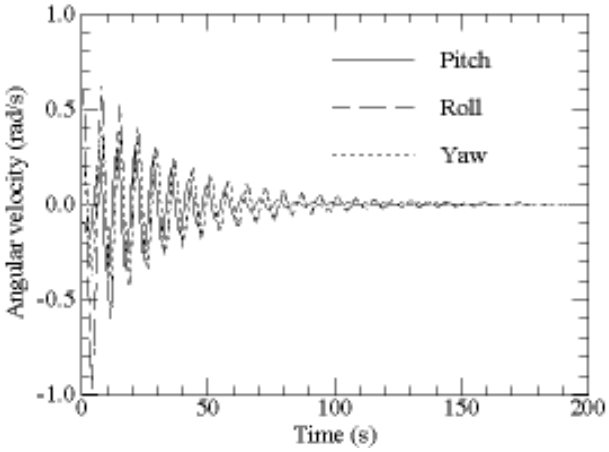

Figura 6: Velocidades angulares com RN de controle.

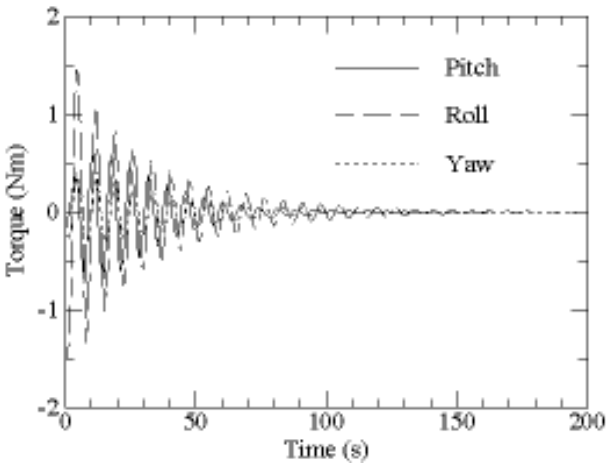

Figura 7: Torque fornecido pela RN de controle.

Note-se que a trajetória de referência, como calculado na Equação 9, baseia-se no erro em posição e velocidade de maneira similar a um controlador PD (proporcional e derivativo). A diferença básica entre eles, no entanto, é que o torque obtido pela RN considera as não-linearidades da dinâmica, enquanto que o controlador PD tem seus ganhos ajustados através de linearizações no modelo dinâmico. Lamentavelmente, mesmo considerando esta aparente vantagem, o desempenho da RN controladora ficou aquém do controlador PD, mostrado na Figura 8. Neste exemplo, o PD atingiu a atitude desejada em apenas 20 segundos, contra mais de 200 segundos da rede. A diferença é ainda mais notável considerando que os ganhos do controlador PD não foram otimizados (adotou-se 0.5 para o ganho proporcional e 7.5 para o derivativo).

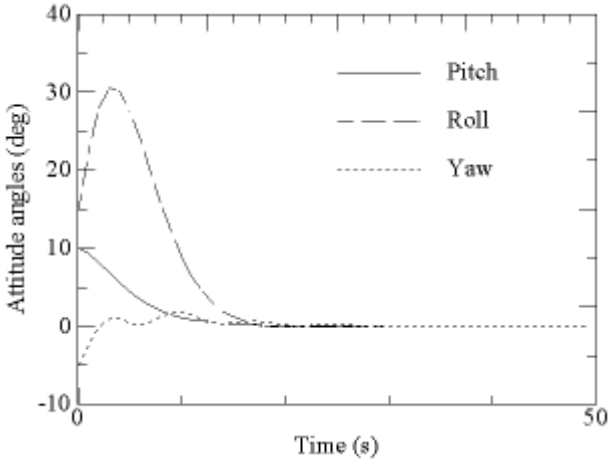

Figura 8: Simulação da atitude com o controlador PD

6. Conclusões

Este trabalho apresentou uma comparação entre um controlador PD convencional e uma rede neural atuando como controlador da atitude de um satélite artificial. Foi mostrado que o desempenho obtido pela RN depende do sistema considerado, e que nem sempre é superior a um controlador PD. Cumpre salientar, no entanto, que as redes neurais têm algumas vantagens intrínsecas que permitem uma exploração mais cuidadosa visando melhorar os resultados. Se, por um lado, ainda não se dispõe de teorias para controles não-lineares, por outro o grande número de parâmetros ajustados empiricamente necessários no treinamento da rede torna o processo extenuante. Há que se considerar, também, que não existem muitos exemplos, na literatura, sobre controle de sistemas com vários graus de liberdade por meio de redes neurais. Como ficou claro, quanto maior o número de variáveis de estado, maior e mais complexa deverá ser a rede. De fato, foi realizada uma outra tentativa de treinar a rede com um torque máximo de u_{max} = 0.15 Nm, mais realista com relação ao tamanho do satélite. No entanto, devido a esse valor, a rede teve que ser treinada com precisão 10 vezes maior do que aquela efetivamente utilizada nas simulações. O treinamento foi tão demorado que teve de ser interrompido quando a rede já contava com 128 neurônios, sem ainda ter atingido a precisão requerida.

Referências

[1] Hunt, K. J.; Sbarbaro, D.; Zbikowski, R.; Gawthrop, P. J. Neural networks for control systems - a survey. *Automatica*, v. 28, n. 6, p. 1083-1112, 1992.

[2] Kawato, M; Uno, Y.; Isobe, M; Susuki, R. Hierarchical neural network model for voluntary movement with application to robotics. *IEEE Control Systems Magazine*, v. 8, n. 2, p. 8-15, Apr. 1988.

[3] Narendra, K. S.; Parthasarathy, K. Identification and control for dynamic systems using neural networks. *IEEE Transactions on Neural Networks*, v. 1, n.1, p. 4-27, Mar. 1990.

[4] Carrara, V. *Redes neurais aplicadas ao controle de atitude de satélites com geometria variável*. São José dos Campos, INPE, Junho 1997 (INPE-6384-TDI/603).

[5] Cybenko, G. Approximation by superposition of a sigmoidal function. *Mathematics of Controls, Signals ans Systems*. v. 2, n. 4, p. 303-314, 1989.

[6] Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, v. 2, n. 5, p. 359-366, 1989.

[7] Nguyen, D. H.; Widrow, B. Neural networks for self-learning control systems. *IEEE Control Systems Magazine*, v. 10, n. 3, p.18-23, Apr. 1990.

[8] Chen, S.; Billings, S. A. Neural networks for nonlinear dynamic system modelling and identification. *International Journal of Control*, v. 56, n. 2, p. 319-346, 1992.

[9] Baffes, P. T.; Shelton, R. O.; Phillips, T. A. *NETS, a neural network development tool*. Huston, Lyndon B. Johnson Space Center, 1991. (JSC-23366)

[10] Billings, S. A.; Jamaluddin, H. B.; Chen, S. Properties of neural networks with applications to modelling non-linear dynamical systems. *International Journal of Control*. v. 55, n. 1, p. 193-224, 1992.

[11] Carrara, V.; Varotto, S. E. C.; Rios Neto, A. Satellite Attitude Control Using Multilayer Perceptron Neural Networks (98-345). *Advances in the Astronautical Sciences*. Vol. 100, Part 1, 1998. p. 565-579.

[12] Hagan, M. T.; Menhaj, M. Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, V. 5, n. 6, pp. 989-993, 1994.

[13] Demuth, H.; Beale, M. Neural Network Toolbox User's Guide Version 3.0. MathWorks, 1997 (in PDF file).