PREDICTIVE CONTROL WITH MEAN DERIVATIVE BASED NEURAL EULER INTEGRATOR DYNAMIC MODEL

Paulo M. Tasinaffo[*]
tasinafo@comp.ita.br

Atair Rios Neto[*]
atairrn@uol.com.br

[*]Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, SP, Brasil

RESUMO

Redes neurais podem ser treinadas para obter o modelo de trabalho interno para esquemas de controle de sitemas dinâmicos. A forma usual adotada é projetar a rede neural na forma de um modelo discreto com entradas atrasadas do tipo NARMA (Non-linear Auto Regressive Moving Average). Em trabalhos recentes a utilização de uma rede neural inserida em uma estrutura de integração numérica tem sido também considerada para a obtenção de modelos discretos para sistemas dinâmicos. Neste trabalho, uma extensão da última abordagem é apresentada e aplicada em um esquema de controle não-linear preditivo (NPC), com uma rede *feed forward* modelando as derivadas médias em uma estrutura de integrador numérico de Euler. O uso de uma rede neural para aproximar a função de derivadas médias, em vez da função de derivadas instantâneas do sistema dinâmico ODE, permite que qualquer precisão desejada na modelagem discreta de sistemas dinâmicos possa ser realizada, com a utilização de um simples integrador Euler, tornando a implementação do controle preditivo uma tarefa mais simples, uma vez que ela somente necessitará lidar com a estrutura linear de um integrador de primeira ordem na determinação das ações de controle. Para ilustrar a efetividade da abordagem proposta, são apresentados resultados dos testes em um problema de transferência de órbitas Terra/Marte e em um problema de controle de atitude em três eixos de satélite comportando-se como corpo rígido.

Artigo submetido em 02/02/2006
1a. Revisão em 14/02/2007
Aceito sob recomendação do Editor Associado
Prof. José Roberto Castilho Piqueira

PALAVRAS-CHAVE: Controle Neural, Controle Preditivo Não-Linear, Redes *Feed forward*, Modelagem Neural de Sistemas Dinâmicos, Integradores Numéricos de Equações Diferenciais Ordinárias.

ABSTRACT

Neural networks can be trained to get internal working models in dynamic systems control schemes. This has usually been done designing the neural network in the form of a discrete model with delayed inputs of the NARMA type (Nonlinear Auto Regressive Moving Average). In recent works the use of the neural network inside the structure of ordinary differential equations (ODE) numerical integrators has also been considered to get dynamic systems discrete models. In this paper, an extension of this latter approach, where a feed forward neural network modeling mean derivatives is used in the structure of an Euler integrator, is presented and applied in a Nonlinear Predictive Control (NPC) scheme. The use of the neural network to approximate the mean derivative function, instead of the dynamic system ODE instantaneous derivative function, allows any specified accuracy to be attained in the modeling of dynamic systems with the use of a simple Euler integrator. This makes the predictive control implementation a simpler task, since it is only necessary to deal with the linear structure of a first order integrator in the calculations of control actions. To illustrate the effectiveness of the proposed approach, results of tests in a problem of orbit transfer between Earth and Mars and in a problem of three-axis attitude control of a rigid body satellite are presented.

KEYWORDS: Neural Control, Nonlinear Predictive Control,

Feed Forward Neural Nets, Dynamic Systems Neural Modeling, Ordinary Differential Equations Numerical Integrators.

1 INTRODUCTION

Multi layer feed forward artificial neural networks have the capacity of modeling nonlinear functions (e.g., Cybenko, 1988; Hornik et al, 1989). This property allows their application in control schemes, where an internal model of the dynamic system is needed, as is for example the case in predictive control (Clarke et al, 1987a, 1987b). A commonly used way of representing the internal model of the dynamics of the system has been to design the neural network to learn a system approximation in the form of a discrete model with delayed inputs of the NARMA type (Non-linear Auto Regressive Moving Average) (Leontaritis and Billings, 1985a, 1985b; Chen and Billings, 1989, 1990 and 1992; Narendra and Parthasarathy, 1990; Hunt et al, 1992; Mills et al, 1994; Liu et al 1998; Norgaard et al, 2000). The neural net designed and trained in this way has the disadvantage of needing too many neurons in the input and hidden layers.

In recent works, the use of a neural ordinary differential equation (ODE) numerical integrator as an approximate discrete model of motion, together with the use of Kalman filtering for calculations of control actions, was proposed and tested in the predictive control of dynamic systems (Rios Neto, 2001; Tasinaffo and Rios Neto, 2003). It was shown and illustrated with tests that artificial feed forward neural networks could be trained to play the role of the dynamic system derivative function in the structure of ODE numerical integrators, to get internal models in nonlinear predictive control schemes. This approach has the advantage of reducing the dimension and complexity of the neural network, and thus of facilitating its training (Wang e Lin, 1998; Rios Neto 2001). It was also shown that the stochastic nature and the good numerical performance of the Kalman filtering parameter estimator algorithm make its choice a good one, not only to train the feed forward neural network (Singhal et al, 1989; Chandran, 1994; Rios Neto, 1997), but also to estimate the predictive control actions (Rios Neto, 2000). Its use allows considering the errors in the output patterns in the supervised training of the artificial neural networks. It also allows the possibility of giving a stochastic meaning to the weight matrices present in the predictive control functional.

This paper further explores the approach of combining feed forward neural networks with the structure of ordinary differential equations (ODE) numerical integrator algorithms to get dynamic systems internal models in predictive control schemes. Instead of approximating the instantaneous derivative function in the dynamic system ODE model, the neural network is used to approximate the mean derivative function (Tasinaffo, 2003). This allows the use of an Euler structure to get a first order neural integrator. In principle this mean derivative based first order neural integrator can provide the same accuracy as that of any higher order integrator. However, it is much simpler to deal with, both in terms of the neural network training and of the implementation of the predictive control scheme.

In what follows, in Section 2 the mathematical foundation, that supports the possibility of getting discrete nonlinear dynamic system models using Euler numerical integrators with mean derivative functions, is presented. In Section 3 it is presented a summary of the method of calculating the discrete control actions in a predictive control scheme with the use of Kalman filtering. In Section 4, results of tests in a problem of orbit transfer between Earth and Mars and in a problem of three - axis attitude control of a rigid body satellite are presented to illustrate the effectiveness of the proposed approach. Finally, in Section 5 a few conclusions are drawn.

2 MEAN DERIVATIVE BASED EULER INTEGRATOR AS A DYNAMIC SYSTEM DISCRETE MODEL

2.1 Fundaments

For the sake of facilitating the understanding and of mathematically supporting the possibility of using a mean derivative based Euler integrator as a dynamic system discrete model, in this section a summary of the results obtained by Tasinaffo (2003) are presented without demonstration. With this purpose, consider the following nonlinear autonomous system of ordinary differential equations,

$$\dot{y} = f(y) \tag{1.a}$$

where,

$$y = [y_1\ y_2\ \ldots\ y_n]^T \tag{1.b}$$

$$f(y) = [f_1(y)\ f_2(y)\ \ldots\ f_n(y)]^T \tag{1.c}$$

Let, by definition, $y_j^i = y_j^i(t)$, **j=1, 2,...,n** be a trajectory, solution of the nonlinear ODE $\dot{y} = f(y)$, starting from $y_j^i(t_o)$at initial time t_o, that belongs to a domain of interest $[y_j^{min}(t_o)\ ,\ y_j^{max}(t_o)]^n$, and where $y_j^{min}(t_o)$ and $y_j^{max}(t_o)$ are finite. It is convenient to introduce the following vector notation to indicate possible initial condition sets and the respective solutions of (1.a):

$$y_o^i = y^i(t_o) = [y_1^i(t_o)\ y_2^i(t_o)\ \ldots\ y_n^i(t_o)]^T \tag{2.a}$$

$$y^i = y^i(t) = [y_1^i(t)\ y_2^i(t)\ \ldots\ y_n^i(t)]^T \tag{2.b}$$

where, **i= 1, 2, ...,** ∞**;** and ∞ is adopted to indicate that the mesh of discrete initial conditions can have as many points as desired.

To start the mathematical background, two important results (e.g., Braun, 1983) about the solution of differential equations (1.a) are considered. The first is about the existence and uniqueness of solutions and the second about the existence of stationary solutions of (1.a).

Theorem 1 (T1) *Let each of the functions* $f_1(y_1, y_2, ..., y_n)$, ..., $f_n(y_1, y_2, ..., y_n)$ *have continuous partial derivatives with respect to* $y_1, ..., y_n$. *Then, the initial value problem* $\dot{y} = f(y), y(t_0)$ *inside a domain of interest* $[y_j^{min}, y_j^{max}]^n$ ***j=1, 2,...,n,*** *in* t_0**,** *has one and only one solution* $y^i = y^i(t)$, *in* R^n, *from each* $y^i(t_0)$ *initial state. If two solutions,* $y = \phi(t)$ *and* $y = \varphi(t)$, *have a common point, then they must be identical.*

Property 1 (P1) *If* $y = \phi(t)$ *is a solution of (1.a), then* $y = \phi(t + c)$ *is also a solution of (1.a), where* ***c*** *is any real constant.*

Since, in general, $\dot{y}^i = f(y^i)$ has not an analytical solution, it is usual to only know a discrete approximation of $y^i = y^i(t)$, in an interval $[t_k , t_{k+n_t}]$, through a set of discrete points, $[y^i(t + k\Delta t)\ y^i[t + (k + 1)\Delta t] ...\ y^i[t + (k + n_t\Delta t)]\ ] \equiv [{}^{k}y^i\ {}^{k+1}y^i\ ...\ ]$, for a given Δ**t**.

By definition, the secant given by two points ${}^{k}y^i$ and ${}^{k+1}y^i$ of the curve $y^i(t)$ is the straight-line segment joining these two points. Thus, from the secants defined by the pair of points ${}^{k}y^i_1$ and ${}^{k+1}y^i_1$, ${}^{k}y^i_2$ and ${}^{k+1}y^i_2$, , ${}^{k}y^i_n$ and ${}^{k+1}y^i_n$ one can define the tangents:

$$\tan_{\Delta t}\alpha^i(t + k\Delta t) = \tan_{\Delta t}{}^{k}\alpha^i = [\ \tan_{\Delta t}{}^{k}\alpha^i_1\ \tan_{\Delta t}{}^{k}\alpha^i_2\ ...\ \tan_{\Delta t}{}^{k}\alpha^i_n\ ]^T \quad (3.a)$$

$$\tan_{\Delta t}{}^{k}\alpha^i_j = \frac{{}^{k+1}y^i_j - {}^{k}y^i_j}{\Delta t} \quad (3.b)$$

Property 2 (P2) *If* ${}^{k}y^i$ *is a discrete solution of* $\dot{y}^i = f(y^i)$ *and* $\Delta t \neq 0, \tan_{\Delta t}{}^{k}\alpha^i$ *exists and is unique.*

Two other important theorems, which relate the values of $\tan^{k}_{\Delta t}\alpha^i$ and $\tan^{k}_{\Delta t}\dot{\alpha}^i$, with the values of the mean derivatives calculated from $[\ {}^{k}y^i\ {}^{k+1}y^i\ ...\ {}^{k+n}y^i]$ and $[\ {}^{k}\dot{y}^i\ {}^{k+1}\dot{y}^i\ ...\ {}^{k+n}\dot{y}^i]$, respectively, are the differential and integral mean value theorems (e.g., Wilson, 1958; Munem et al, 1978; Sokolnikoff et al, 1966), enunciated in what follows.

Theorem 2 (T2) *(Differential mean value theorem): If a function* $y^i_j(t)$, *for* ***j=1,2,...,n,*** *is defined and continuous in the closed interval* $[t_k, t_{k+1}]$ *and is differentiable in the open interval* (t_k, t_{k+1}), *then there is at least one* t^*_k, $t_k < t^*_k < t_{k+1}$ *such that*

$$\dot{y}^i_j(t^*_k) = \frac{{}^{k+1}y^i_j - {}^{k}y^i_j}{\Delta t} \quad (4)$$

T2 assures that given a secant of a differentiable $y^i(t)$ it is always possible to find a point between ${}^{k+1}y^i$ and ${}^{k}y^i$ of the intersection of the secant with the curve in t_k and t_{k+1}**,** such that the tangent to this intermediate point is parallel to the secant. The value $\dot{y}^i(t^*_k)$ is called the *mean derivative* of $y^i(t)$ in $[t_k, t_{k+1}]$**.**

Theorem 3 (T3) *(Integral mean value theorem): If a function,* $y^i_j(t)$, *for* ***j=1, 2,..., n*** *is continuous in the closed interval* $[t_k, t_{k+1}]$, *then there exists at least one* t^x_k *interior to this interval, such that*

$$y(t^x_k) \cdot \Delta t = \int_{t_k}^{t_{k+1}} y^i(t) \cdot dt \quad (5)$$

In general t^*_k and t^x_k are different and it is important to notice that the theorems do not tell how to determine these points.

Property 3 (P3) *The mean derivative* $\dot{y}^i(t^*_k)$ *of* $y^i(t)$ *in the closed interval* $[t_k, t_{k+1}]$ *is equal to* $\tan_{\Delta t}{}^{k}\alpha^i$, *as an immediate consequence of the definition of mean derivatives.*

Theorem 4 (T4) *The point* ${}^{k+1}y^i_j$ *of the solution of the system of nonlinear differential equations* $\dot{y}^i = f(y^i)$, *for* ***j=1, 2,...,n,*** *can be determined through the relation* ${}^{k+1}y^i_j = \tan^{k}_{\Delta t}\alpha^i \cdot \Delta t + {}^{k}y^i_j$ *for a given* ${}^{k}y^i$ *and* Δt.

Corollary 1 (C1)- *The solution of the system of nonlinear differential equations* $\dot{y}^i = f(y^i)$, *at a given discrete point,* ${}^{k+m}y^i_j$, *for* ***j=1, 2,...,n,*** *can be determined, given an initial* ${}^{k}y^i$, *by the relation:*

$$ {}^{k+m}y^i_j = \sum_{l=0}^{m\ -\ 1} \tan^{k+l}_{\Delta t}\alpha^i \cdot \Delta t + {}^{k}y^i_j \quad (6)$$

Corollary 2 (C2) *For the system,* $\dot{y}^i = f(y^i)$, *the following relation is valid:* $\tan^{k}_{m\cdot\Delta t}\alpha^i_j = \frac{1}{m} \cdot \sum_{l=0}^{m\ -\ 1} \tan^{k+l}_{\Delta t}\alpha^i_j$, *for* ***j=1, 2,...,n****.*

Notice that for the situation where the system of Eq. (1a) is autonomous, $y^i1(t_1) = y^i2(t_2)$ for $i_1 \neq i_2$ and $t_1 \neq t_2$ implies that $\dot{y}^i1(t_1) = \dot{y}^i2(t_2)$. This property establishes that two trajectories of $\dot{y} = f(y)$ starting from two different initial conditions, $y^i1(t_0)$ and $y^i2(t_0)$, for $i_1 \neq i_2$, will have the same derivatives only if $y^i1(t_1) = y^i2(t_2)$**,** even when $t_1 \neq t_2$.

The question remaining is if the mean derivative $\tan^k_{\Delta t}\alpha^i$ of the interval $[{}^ky^i, {}^{k+1}y^i]$ is also autonomous, that is, time invariant? The properties that follow answer this question.

Property 4 (P4) *If* $y^i1(t)$ *and* $y^i2(t)$ *are solutions of* $\dot{y} = f(y)$ *starting from* $y^i1(t_0 = 0)$ *and* $y^i2(t_0 = 0)$*, respectively, and if* $y^i1(t_0 = 0) = y^i2(T)$ *for* $\boldsymbol{T>0}$*, then* $y^i1(\Delta t) = y^i2(T + \Delta t)$ *for any given* Δt.

Property 5 (P5) *If* $y^i1(t_1) = y^i2(t_2)$*, for* $i_1 \neq i_2$ *and* $t_1 \neq t_2$**,** *then,* $\tan_{\Delta t}\alpha^i1(t_1) = \tan_{\Delta t}\alpha^i2(t_2)$ *for* $\Delta t > 0$*, that is,* $\tan^k_{\Delta t}\alpha^i1$ *is autonomous.*

This result is useful since it determines that it is enough to know the values of $\tan^k_{\Delta t}\alpha^i$, for $i = 1, 2, ..., \infty$ at t_0, in a region of interest $[y^{min}_j, y^{max}_j]^n$, **j=1, 2,...,n**, because for $\mathbf{t>t}_0$ they will repeat, as long the boundaries of $[y^{min}_j, y^{max}_j]^n$ are observed.

Notice also that the trajectories of the dynamic system when propagated ahead will have angles ${}^k\alpha(i)$ varying only in the interval $-\frac{\pi}{2} < {}^k\alpha(i) < \frac{\pi}{2}$, which will thus be unique. When retro propagated, $\frac{\pi}{2} < {}^k\alpha(i) < \frac{3\cdot\pi}{4}$, and thus ${}^k\alpha(i)$ will also be unique in this case.

Theorem 5 (T5) *The result of T4 is still valid when discrete values of control* ${}^ku$ *in each* $[t_k, t_{k+1}]$ *are used to solve the dynamic system:*

$$\dot{y}^i = f(y^i, u) \tag{7}$$

Demonstration: In this case the continuous function, $f(y^i,u)$, with ${}^ku$ approximated as constant in $[t_k, t_{k+1}]$, can be viewed as parameterized with respect to the control variable and, thus, for any discrete interval the existence of the mean derivative $\dot{y}^i(t^*_k) = \frac{{}^{k+1}y^i - {}^ky^i}{\Delta t} = \tan_{\Delta t}{}^k\alpha^i$ is guaranteed and the result in Eq. (6) is still valid.

2.2 Numerical Integrators with Neural Mean Derivatives to Represent Dynamic Systems

Consider the capacity of a feed forward neural network to approximate nonlinear functions (e.g., Zurada, 1992). From the previous section, one can conclude that it is possible to have a neural network to learn the mean derivatives of a given dynamical system and use them in an ODE Euler integrator structure to get a discrete representation of this system. In the proposed approach, a first possibility was adopted as illustrated in Fig. 1, where the neural network is trained to directly learn the dynamic system mean derivative, which is then inserted in the structure of the Euler numerical integrator. In this scheme, the neural network is trained to learn the function of mean derivatives from the sampled input values of state ${}^ky^i$ and control ${}^ku$, with a previously fixed discrete interval Δt The value of the training output pattern $\tan^k_{\Delta t}\alpha^i = \frac{{}^{k+1}y^i - {}^ky^i}{\Delta t}$ is generated with the help of a numerical integrator of high order used to simulate one step ahead with negligible errors ${}^{k+1}y^i$, the solution of the system $\dot{y}^i = f(y^i,u)$.

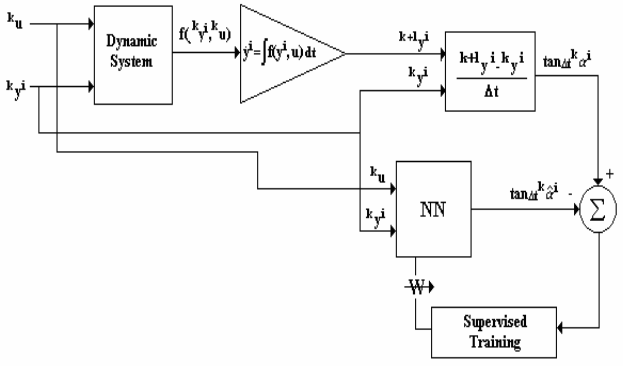

Figure 1: Supervised Training of Mean Derivatives of $\dot{y}^i = f(y^i,u)$.

A second possibility that could be used, based on that adopted by Wang and Lin (1998), is depicted in Fig. 2. It is one where using the outputs of an Euler integrator the neural network is indirectly trained to learn the dynamic system mean derivative. In this case, ${}^{k+1}\hat{y}^i$, the value of state estimated by the neural Euler numerical integrator, is the output value compared to the training pattern ${}^{k+1}y^i$ to generate the error signal for the supervised training. The neural network is trained to learn the function of mean derivatives $\tan^k_{\Delta t}\alpha^i \equiv \tan_{\Delta t}\alpha^i({}^ky,{}^ku)$ from the values of state ${}^ky^i$, control ${}^ku$ and of a previously fixed discrete intervalΔt. In Fig. 2, ${}^{k+1}y^i$ is the value of the training pattern generated off line by a numerical integrator of high order used

to simulate the system $\dot{y}^i = f(y^i,u)$, and ${}^{k+1}\hat{y}^i$ is the value the neural network tries to estimate for ${}^{k+1}y^i$, in the supervised training. The relation of recurrence between ${}^{k}y^i$ and ${}^{k+1}y^i$ is expressed by ${}^{k+1}y^i = \tan^k_{\Delta t}\alpha^i \cdot \Delta t + {}^{k}y^i$, where $\tan^k_{\Delta t}\alpha^i \cong \hat{f}({}^{k}y^i, {}^{k}u, \hat{w})$ is the mean derivative to be approximated by the neural network. It should be noticed that if ${}^{k+1}y^i$ is obtained from the Euler integration, then $\tan^k_{\Delta t}\alpha^i$ converges to the function of derivatives $\dot{y}^i = f(y^i,u)$. But if ${}^{k+1}y^i$ is obtained from the use of a high order integrator or experimentally, then $\tan^k_{\Delta t}\alpha^i$ will converge to the function of mean derivatives.

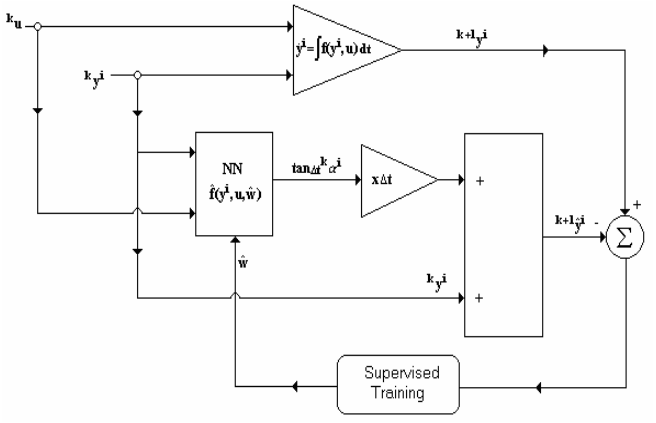

Figure 2: Supervised Training of an Euler Neural Integrator Using Mean Derivatives of. $\dot{y}^i = f(y^i,u)$

A correspondent algorithm to get the mean derivative based first order neural integrator would then be as follows:

1. Given in $\mathbf{t}_o$ the domains of interest $[y^{min}_j(t_o), y^{max}_j(t_o)]^{n_1}$, **j=1, 2, ..., $\mathbf{n}_1$**, and $[u^{min}_k, u^{max}_k]^{n_2}$, **k=1, 2, ..., $\mathbf{n}_2$**, generate inside these domains, **m** uniformly distributed random vectors to be the input p_i, **i=1, 2, ..., m,** of training patterns to the feed forward neural network.

2. Employing a high order ODE numerical integrator, propagate ahead with step size Δt the inputs p_i, **i=1, 2, ..., m** , generating the state vectors , **i=1, 2, ..., m**, at $t_o + \Delta t$.

3. Calculate the vectors $\mathbf{T}_i$ to be used as training output patterns:

$$T_i = \frac{1}{\Delta t} \cdot [y^i_1(t_o + \Delta t) - y^i_1(t_o)\ y^i_2(t_o + \Delta t) - y^i_2(t_o) \cdots y^i_{n_1}(t_o + \Delta t) - y^i_{n_1}(t_o)]^T = [\tan^k_{\Delta t}\alpha^i_1\ \tan^k_{\Delta t}\alpha^i_2 \cdots \tan^k_{\Delta t}\alpha^i_{n_1}]^T = \left(\tan^k_{\Delta t}\alpha^i\right)^T \quad (8)$$

Notice that since the function $\tan^k_{\Delta t}\alpha^i$ is also autonomous it is only necessary propagate ahead p_i, **i=1, 2, ..., m,** with step size Δt, to get the neural network output patterns $\mathbf{T}_i$.

4. Do the supervised training of the neural network, using the patterns **{(p_i, $\mathbf{T}_i$)}**.

5. After training the neural network with a specified accuracy, there results the dynamic system discrete model, in the form of a mean derivative based first order neural integrator:

$${}^{k+1}y^i = \tan^k_{\Delta t}\alpha^i \cdot \Delta t + {}^{k}y^i \quad (9)$$

Notice that using the scheme of Fig. 1, with $T_i = \tan^k_{\Delta t}\alpha^i$ as output patterns, avoids calculating the back propagation with ${}^{k+1}y^i = \tan^k_{\Delta t}\alpha^i \cdot \Delta t + {}^{k}y^i$.

To analyze the local error of this neural Euler integrator, consider the exact value ${}^{k+1}y^i$ and the estimated value ${}^{k+1}\hat{y}^i$, respectively given by Eqs. (10) and (11).

$${}^{k+1}y^i = \tan^k_{\Delta t}\alpha^i \cdot \Delta t + {}^{k}y^i \quad (10)$$

$${}^{k+1}\hat{y}^i \cong (\tan^k_{\Delta t}\alpha^i + e_m) \cdot \Delta t + {}^{k}y^i \quad (11)$$

where $\mathbf{e}_m$ is the error in the output of the neural network trained to learn the function of mean derivatives $\tan^k_{\Delta t}\alpha^i_j$ inside a domain of interest. Due to the capacity of approximation of the neural network, this error can be less than any specified value. Thus, ${}^{k+1}\hat{y}^i$, in Eq. (11), can have the desired accuracy, since for a fixed $\Delta t > 0$ the neural network is approximating inside a domain of interest the function of mean derivatives $\tan^k_{\Delta t}\alpha^i_j$ that is invariant in time, and e_m can be made as small as specified.

Figure 3 better illustrates this situation. Consider ${}^{k+1}y^i$, ${}^{k+1}y^i_a$ and ${}^{k+1}y^i_e$, respectively representing the exact value of the solution of $\dot{y}^i = f(y^i,u)$ at t_{k+1}, the approximate value of ${}^{k+1}y^i$ obtained from a high order numerical integrator, and the approximate value of ${}^{k+1}y^i$ obtained from an Euler integrator. As indicated by Fig. 3, if it is taken ${}^{k+1}y^i = {}^{k+1}y^i_e$, in the scheme of Fig. 2, then $\tan^k_{\Delta t}\alpha^i = \hat{f}({}^{k}y^i, {}^{k}u, \hat{w}) \cong f({}^{k}y^i, {}^{k}u)$, but if it is taken ${}^{k+1}y^i = {}^{k+1}y^i_a$, and if ${}^{k+1}y^i_e$ is away from ${}^{k+1}y^i$, then $\tan^k_{\Delta t}\alpha^i$ during the phase of training will approximately converge to the function of mean derivatives instead of converging to $f(y^i,u)$.

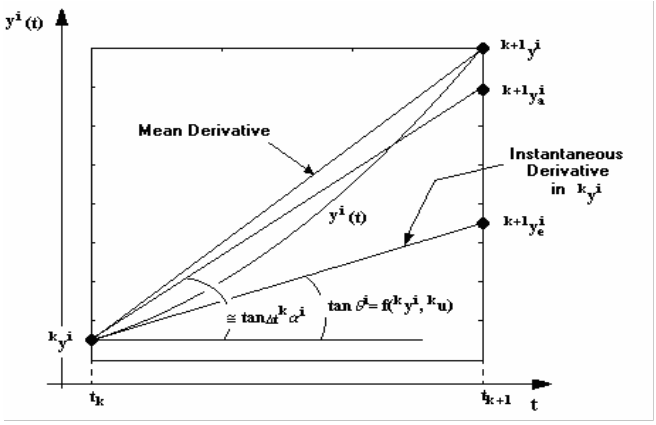

Figure 3: Representation of mean $\tan^{k}_{\Delta t}\alpha^{i}$ **and instantaneous** $\tan^{k}\theta^{i} = f(^{k}y^{i},^{k}u)$ **derivatives, for** $\dot{y}^{i} = f(y^{i},u)$

3 NEURAL PREDICTIVE CONTROL SCHEME

The neural predictive control scheme presented in what follows was proposed and demonstrated by Rios Neto (2000). In a problem of neural predictive control of a dynamic system (Mills et al, 1994), it adopts a heuristic and theoretical approach to solve the problem of minimizing a quadratic functional subject to the constraint of a neural network predictor, representing the dynamics of the system to be controlled. In the proposed scheme, the problems of training the neural network and of determining the predictive control actions are seen and treated in an integrated way, as problems of stochastic optimal linear estimation of parameters.

The problem to be solved is that of controlling a dynamical system modeled by an ODE:

$$\dot{y} = f(y) \tag{12}$$

It is assumed that the system to be controlled can be approximated by a discrete model. That is, for $t_j = t + j \cdot \Delta t$:

$$y(t_j) \cong f[y(t_{j-1}), ..., y(t_{j-n_y}); u(t_{j-1}), ..., u(t_{j-n_u})] \tag{13}$$

where, $y(t), ..., y(t_{1-n_y})$ and $u(t_{-1}), ..., u(t_{1-n_u})$ are the past system responses and control actions, respectively.

In the usual neural predictive control scheme, a feed forward neural network is trained to learn a discrete model as in Eq. (13). This model is then used as an internal system response model to get the smooth control actions that will track a reference response trajectory by minimizing (e.g., Clarke et al, 1987a; Clarke et al, 1987b; Liu et al, 1998) the finite horizon functional:

$$J = [\sum_{j=1}^{n_h} [y_r(t_j) - \hat{y}(t_j)]^T \cdot r_y^{-1}(t) \cdot [y_r(t_j) - \hat{y}(t_j)] + \sum_{j=0}^{n_h - 1} [u(t_j) - u(t_{j-1})]^T \cdot r_u^{-1}(t) \cdot [u(t_j) - u(t_{j-1})]] / 2 \tag{14}$$

where, $y_r(t_j)$ is the reference response trajectory at time $\mathbf{t}_j$; $\mathbf{n}_h$ is the number of steps in the finite horizon of optimization; $r_r^{-1}(t_j)$ and $r_u^{-1}(t_j)$ positive definite weighting matrices; $\hat{y}(t_j)$ is the output of the feed forward neural network previously trained to approximate a discrete model of the dynamic system response.

The determination of the predictive control actions can be treated as a parameter estimation problem, if the minimization of the functional of Eq. (14) is seen as the following stochastic problem:

$$y_r(t_j) = \hat{y}(t_j) + v_y(t_j) \tag{15.a}$$

$$0 = u(t_{j-1}) - u(t_{j-2}) + v_u(t_{j-1}) \tag{15.b}$$

$$E[v_y(t_j)] = 0, E[v_y(t_j) \cdot v_y^T(t_j)] = r_y(t_j) \tag{15.c}$$

$$E[v_u(t_j)] = 0, E[v_u(t_j) \cdot v_u^T(t_j)] = r_u(t_j) \tag{15.d}$$

with, $j = 1, 2, ..., n_h$; where $\hat{y}(t_j) = f[\hat{y}(t_{j-1}), ..., \hat{y}(t_{j-n_y}); u(t_{j-1}), ..., u(t_{j-n_u}),w]$ are the outputs of the neural network which is recursively used as a predictor of the dynamic system responses in the horizon of optimization and it is understood that for $t_{j-k} \leq t$, $\hat{y}(t_{j-k})$ are estimations or measurement of already occurred values of outputs, in the control feedback loop; $v_y(t_j)$ and $v_u(t_j)$ are the uncorrelated noises for different values of t_j.

To solve the problem of Eqs.(15) an iterative approach is needed, where in each ith iteration a perturbation is done to get a linear approximation of Eq. (15.a):

$$\alpha(i) \cdot [y_r(t_j) - \bar{y}(t_j,i)] = \sum_{k=0}^{j-1} [\partial\hat{y}(t_j) \big/ \partial u(t_k)]_{\{\bar{u}(t_k;i)\}} \cdot [u(t_k,i) - \bar{u}(t_k,i)] + v_y(t_j) \tag{16}$$

where **k** starts at zero, even for $\mathbf{j} > \mathbf{n}_u$, as a consequence of $\hat{y}(t_j)$ recursively being a function of $u(t_{j-2}), ..., u(t)$ through the successive recursions starting with $\hat{y}(t_{j-1}), ..., \hat{y}(t_{j-n_y})$ (see the Appendix, for details about the recurrence relations needed in the calculations of the partial derivatives, for

the proposed case where discrete nonlinear dynamic system models using Euler numerical integrators with mean derivative functions are used); $0 < \alpha(\text{i}) \leq 1$ is a parameter to be adjusted to guarantee the hypothesis of linear perturbation; and the partial derivatives are calculated either by numerical differentiation or by using the chain rule to account for the composed function situation, including the back propagation rule (see, e.g., Chandran (1994)) in the feed forward neural network that approximates the derivative function of the dynamic system.

The formulation as a stochastic linear estimation problem in each iteration is complete if the recursion in Eq.(15.b) is taken in account:

$$\alpha(\text{i}) \cdot [\hat{\text{u}}(\text{t}\_1) - \bar{\text{u}}(\text{t}_\text{l},\text{i})] = \quad (17a)$$
$$[\text{u}(\text{t}_\text{l},\text{i}) - \bar{\text{u}}(\text{t}_\text{l},\text{i})] + \sum_{\text{k}=0}^{\text{l}} \text{v}_\text{u}(\text{t}_\text{k})$$
$$\text{l} = 0, 1, ..., \text{n}_\text{h} - 1; \quad \text{i} = 1, 2, ..., \text{I} \quad (17b)$$

In a more compact notation:

$$\text{U}_\text{l}(\text{t,i}) \equiv \text{u}(\text{t}_\text{l},\text{i}) \quad (18.a)$$

$$\hat{\text{U}}_\text{l}(\text{t}\_1) \equiv \hat{\text{u}}(t\_1) \quad (18.b)$$

$$\alpha(i)\left[\hat{\mathbf{U}}(t_{-1}) - \bar{\mathbf{U}}(t,i)\right] = \mathbf{U}(t,i) - \bar{\mathbf{U}}(t,i) + \mathbf{V}_u(t) \quad (18.c)$$

$$\alpha(i)\,\bar{\mathbf{Z}}^u(t,i) = \mathbf{H}^u(t,i)\left[\mathbf{U}(t,i) - \bar{\mathbf{U}}(t,i)\right] + \mathbf{V}_y(t) \quad (18.d)$$

where the meaning of compact variables become obvious by comparison with Eqs.(16) and (17). Applying the Kalman filtering algorithm, the following solution results in a typical iteration (Rios Neto, 2000):

$$\hat{\text{U}}(t,i) = \bar{\text{U}}(\text{t,i}) + \alpha(\text{i}) \cdot [\hat{\text{U}}(\text{t}\_1) - \bar{\text{U}}(\text{t,i})] + \text{k}(\text{t,i}) \cdot \alpha(\text{i}) \cdot [\bar{\text{Z}}^\text{u}(\text{t,i}) - \text{H}^\text{u}(\text{t,i}) \cdot [\hat{\text{U}}(\text{t}\_1) - \bar{\text{U}}(\text{t,i})]] \quad (19.a)$$

$$\text{k}(\text{t,i}) = \text{R}_\text{u}(\text{t}) \cdot \text{H}^{\text{u}^\text{T}}(\text{t,i}) \cdot [\text{H}^\text{u}(\text{t,i}) \cdot \text{R}_\text{u}(\text{t}) \cdot \text{H}^{\text{u}^\text{T}}(\text{t,i}) + \text{R}_\text{y}(\text{t})]^{-1} \equiv [\text{R}_\text{u}^{-1}(\text{t}) + \text{H}^{\text{u}^\text{T}}(\text{t,i}) \cdot \text{R}_\text{y}^{-1}(\text{t}) \cdot \text{H}^\text{u}(\text{t,i})]^{-1} \cdot \text{H}^{\text{u}^\text{T}}(\text{t,i}) \cdot \text{R}_\text{y}^{-1}(\text{t}) \quad (19.b)$$

$$\bar{\text{U}}(t, i+1) = \hat{\text{U}}(t,i); \alpha(\text{i}) \leftarrow \alpha(\text{i}+1); \hat{\text{U}}(t) = \hat{\text{U}}(t, I) \quad (19.c)$$

$$\hat{\text{R}}_\text{u}(\text{t,I}) = [\text{I}_\text{u} - \text{K}(\text{t,I}) \cdot \text{H}^\text{u}(\text{t,I})] \cdot \text{R}_\text{u}(\text{t}) \quad (19.d)$$

where, $\text{i} = 1, 2, ..., \text{I}$; $\text{R}_\text{u}(\text{t})$, $\text{R}_\text{y}(\text{t})$ and $\text{R}(\hat{t}, I)$ are the covariance matrices of $\text{V}_\text{u}(\text{t})$, $\text{V}_\text{y}(\text{t})$ and $(\hat{\text{U}}(\text{t,I}) - \text{U}(\text{t}))$, respectively; and I_u is an identity matrix.

A correspondent algorithm for this predictive control scheme in a typical time step **t** would then be as follows.

1. The control $\hat{u}(t_{-1})$ (see Eq. (18b)) is the estimated solution from the last control step. In the ith iteration: the approximated estimated value of control is $\text{U}(\bar{t}, i) = \text{U}(t, \hat{i} - 1)$; $\alpha(\text{i}) \leftarrow \alpha(\text{i} - 1)$; and for **i=1** estimates or extrapolations of estimates of last control step are used.

2. Calculate the partial derivatives $\frac{\partial \hat{\text{y}}(t_\text{j})}{\partial \text{u}(\text{t}_\text{k})}$ of Eq.(16), using the expressions of Eqs. (1A) to (3A) of the Appendix. get $\text{H}^\text{u}(\text{t,i})$ and $\bar{\text{Z}}^\text{u}(\text{t,i})$, in Eq. (18c).

3. Estimate $\hat{\text{U}}(\hat{t}, i)$ with the Kalman filtering of Eqs. (19.a), (19.b). Notice that the Kalman filtering can be done either in batch or sequentially, by recursively processing component to component, in sequence. Increment **i**, and repeat steps, until the $\hat{\text{y}}(t_\text{j})$ are sufficiently close to $\text{y}_\text{r}(\text{t}_\text{j})$ according to a specified error, usually taken $3 \cdot \sigma$ of v_y, and when this occurs take:

$$\hat{\text{U}}(t) = \text{U}(\hat{t}, I) \quad (20)$$

4 NUMERICAL TESTS

4.1 Tests in an Earth Mars Orbit Transfer.

This is a problem of low thrust orbit transfer where the state variables are the rocket mass m, the orbit radius r, the radial speed w and the transversal speed v, and where the control variable is the thrust steering angle θ, measured from local horizontal. The ODE (e.g., Sage, 1968) of this dynamic system are:

$$\dot{\text{m}} = -0.0749 \quad (21.a)$$

$$\dot{\text{r}} = \text{w} \quad (21.b)$$

$$\dot{\text{w}} = \frac{\text{v}^2}{\text{r}} - \frac{\mu}{\text{r}^2} + \frac{\text{T}\cdot\sin\theta}{\text{m}} \quad (21.c)$$

$$\dot{\text{v}} = \frac{-\text{w}\cdot\text{v}}{\text{r}} + \frac{\text{T}\cdot\cos\theta}{\text{m}} \quad (21.d)$$

where the variables have been normalized with:$\mu = 1.0$, the gravitational constant; $T = 0.1405$, the thrust; with $t_o = 0$ and $t_f = 5$, initial and final times, where each unit of time is equal to 58.2 days. The predictive control is used on line to produce control actions that make the spacecraft follow a reference trajectory determined off line.

Initially tests were conducted to evaluate the neural integrator capacity of giving an accurate discrete model of the dy-

namics. To approximate the vector mean derivative function, a multiplayer perceptron feed forward neural network, with 41 neurons (this number of neurons can be determined only empirically) with the hyperbolic tangent as activation functions ($\lambda = 2$), in the hidden layer, with identity activation, in the output layer, and with input bias in the hidden and output layers, was used. This feed forward neural net was trained with a parallel processing Kalman filtering algorithm (e.g., Singhal, 1989; Rios Neto, 1997), with 3600 input – output training patterns until a mean square error of $2.4789.10^{-6}$ was reached and tested with 1400 patterns, reaching a mean square testing error of $2.7344.10^{-6}$.

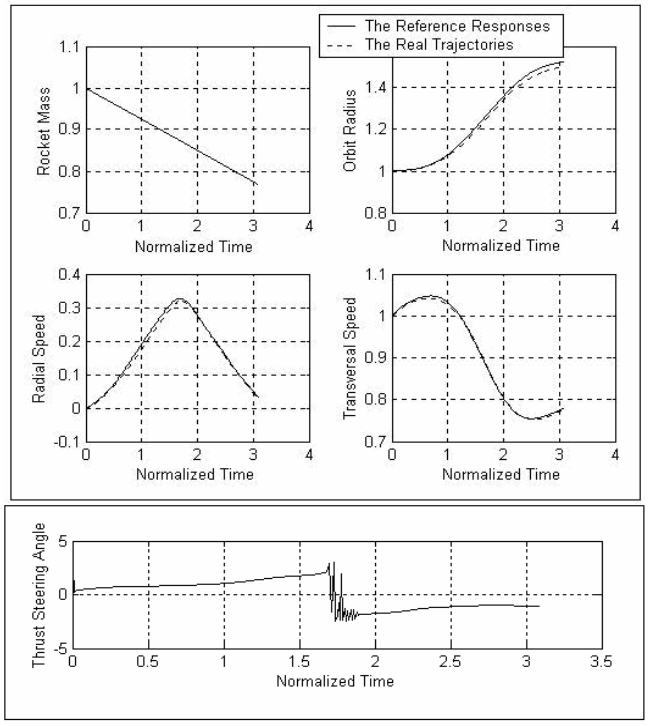

Figure 4: Predictive Control with Mean Derivatives Based Neural Euler Integrator Dynamic Model in an Earth Mars Orbit Transfer ($\Delta t = 0.01, nh = 1$).

This mean derivative neural network was then used in an Euler integrator structure to produce an internal model of the orbit transfer problem dynamics to be used in a predictive control scheme where the reference was defined as the optimal minimum time transfer trajectory. Results obtained are shown in Figs. 4 and 5, for a discrete step size of 0.01 of normalized time (0.582 days) and receding horizons of 1 and 5 steps ahead, respectively. These results illustrate the effectiveness of the proposed approach when applied in this kind of problem.

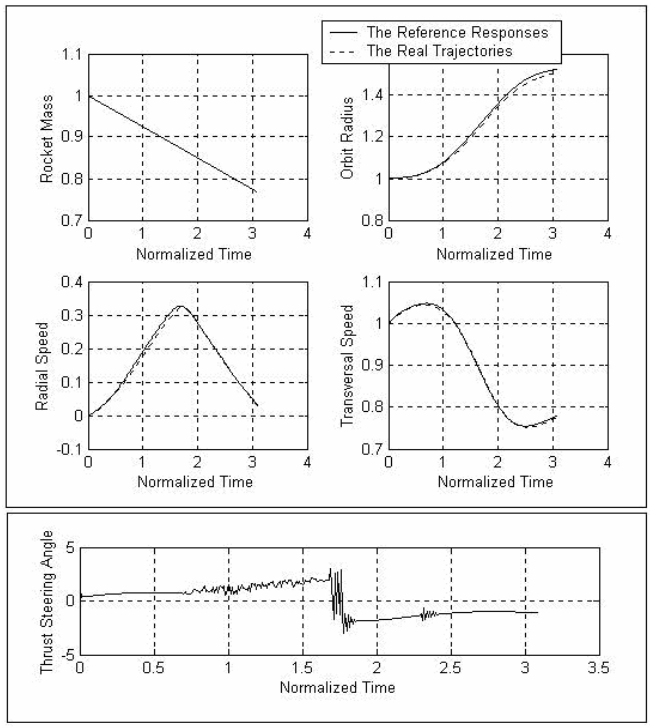

Figure 5: Predictive Control with Mean Derivatives Based Neural Euler Integrator Dynamic Model in an Earth Mars Orbit Transfer (Δt=0.01, n_h=5).

4.2 Tests in a Three-Axis Satellite Attitude Control

In this case, the attitude control in three axes of a rigid body satellite is considered, with the correspondent dynamic equations given as follows (Wertz, 1978; Kaplan, 1976):

$$\begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\varphi} \end{Bmatrix} = \frac{1}{\cos\theta} \cdot \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \cos\varphi.\cos\theta & \sin\varphi.\cos\theta & 0 \\ -\cos\varphi.\sin\theta & -\sin\varphi.\sin\theta & \cos\theta \end{bmatrix} \cdot \begin{Bmatrix} w_x \\ w_y \\ w_z \end{Bmatrix} \tag{22.a}$$

$$\begin{aligned} \dot{w}_X &= \left(\frac{I_y - I_z}{I_x}\right) \cdot w_y.w_z + \frac{T_x}{I_x} \\ \dot{w}_Y &= \left(\frac{I_z - I_x}{I_y}\right) .w_x.w_z + \frac{T_y}{I_y} \\ \dot{w}_Z &= \left(\frac{I_x - I_y}{I_z}\right) .w_x.w_y + \frac{T_z}{I_z} \end{aligned} \tag{22.b}$$

where, ϕ, θ and φ are the Euler angles; I_x, I_y, I_z are the principal moments of inertia; $\boldsymbol{w}_x$,$\boldsymbol{w}_y$ and $\boldsymbol{w}_z$ the angular velocity components, in the principal body axes; $\boldsymbol{T}_x$,$\boldsymbol{T}_y$ and $\boldsymbol{T}_z$ the control torques. The reference trajectories are defined such as to drive the Euler angles asymptotically to the origin, using updating data from navigation (Silva, 2001):

$$\phi_{ref}(t_{k+1}) = \phi(t_k) \cdot \exp\left[-\beta \cdot (t_{k+1} - t_k)\right] \qquad (23.a)$$

$$\theta_{ref}(t_{k+1}) = \theta(t_k) \cdot \exp\left[-\beta \cdot (t_{k+1} - t_k)\right] \qquad (23.b)$$

$$\varphi_{ref}(t_{k+1}) = \varphi(t_k) \cdot \exp\left[-\beta \cdot (t_{k+1} - t_k)\right] \qquad (23.c)$$

Initially, tests were conducted to evaluate the neural integrator capacity of giving an accurate discrete model of the attitude dynamics. To approximate the vector mean derivative function, a multiplayer perceptron feed forward neural network, with 20 neurons with the hyperbolic tangent as activation functions ($\lambda = 1$), in the hidden layer, neurons with identity activation, in the output layer, and with input bias in the hidden and output layers, was used. This feed forward neural net was also trained with a parallel processing Kalman filtering algorithm, with 3200 input – output training patterns until a mean square error of $8.621.10^{-5}$ was reached and tested with 800 patterns, reaching a mean square testing error of $8.669.10^{-5}$.

This mean derivative neural network was then used in an Euler integrator structure to produce the internal model of the attitude dynamics to be used in the predictive control scheme with the reference as defined in Eqs.(23) and testing data as given in Table 1. Results obtained are shown in Fig. 6.

Table 1: Testing Data

Initial Conditions: ϕg $=$ gg $=$ gg**=**g**2**w^o ω_xg $=$ gω_yg $=$ gω_zg $=$ g**1[rpm]=**$\frac{\pi}{30} \cdot 1$**[rad/S]**	**Moments of Inertia:** $I_x = 30, I_y = 40, I_z = 20$
t$_o$= 0 [s] e t$_f$= 400 [s]	$\beta = 0.05$,**R$_u$=10^{-2} e R$_y$=10^{-6}**
Δt$_1$ (Neural Integrator step size) = 0.4 [s]	**Δt$_3$(Validation Model step size) =** $\frac{2}{50}$**[s] Fourth Order Runge-Kutta**
Δt$_2$ (Predictive Control Horizon) = 2 [s]	**Linear Perturbation Parameter (Eq. 16)** $\alpha = 0.15$

These results illustrate the effectiveness of the proposed approach when applied to this kind of problem. The oscillatory behavior in the y component of angular velocity may be due to the fact that the reference trajectory did not include explicitly the derivative terms corresponding to the angular regulation.

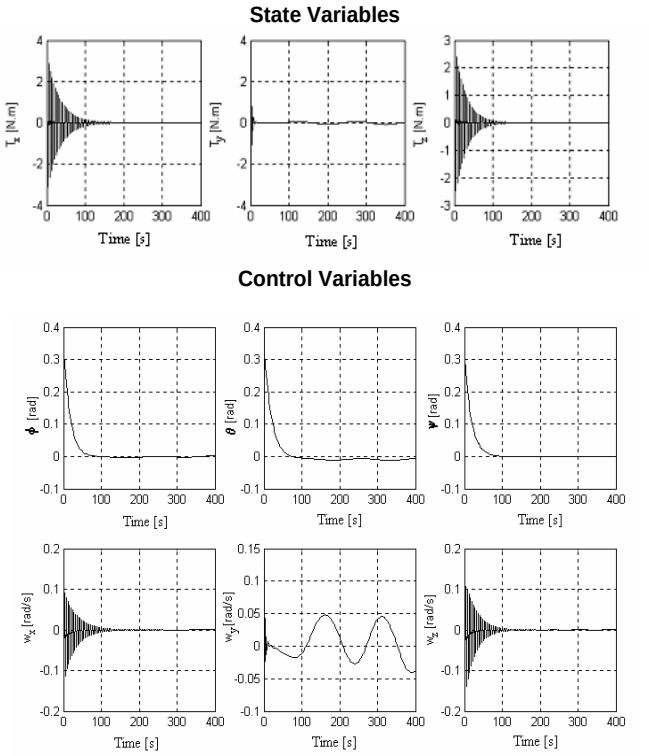

Figure 6: Predictive Control with Mean Derivatives Based Neural Euler Integrator Dynamic Model in a Satellite Three-Axis Attitude Control (Δ**t=2 [s];** $\mathbf{n}_h$).

5 CONCLUSIONS

A new approach of predictive control, using a mean derivative based neural Euler integrator as the internal dynamic system model, was presented. The structure of an ODE Euler numerical integrator was used to get neural discrete forward models where the neural network has only to learn and approximate the algebraic and static mean derivative function in the dynamic system ODE.

The tests indicate the effectiveness of using the mean derivative based neural model of the dynamic system as an internal model in the control scheme and reinforced the expected following characteristics:

- It is a simpler task to train a feed forward neural network to learn the algebraic, static function of the dynamic system ODE mean derivatives (where the inputs are samples of state and control variables), than to train it to learn a NARMA type of discrete model (where the inputs are samples of delayed responses and controls).
- The neural network in the neural ODE integrator results to be simpler, in terms of the necessary number of layers

and number of neurons, since it does not have to learn the dynamic law, but only the derivative function.

The use of a Kalman filtering based approach to get the neural predictive control actions led to results where:

- The stochastic interpretation of errors gave more realism in the treatment of the problem, and facilitated the adjustment of weight matrices in the predictive control functional.
- The local parallel processing version of the Kalman filtering algorithm used in the control scheme exhibited efficiency and efficacy equivalent to that of the correspondent neural network training Kalman filtering algorithm. This was expected, since they are completely similar algorithms used to solve numerically equivalent parameter estimation problems.
- Only one step ahead was sufficient in the receding horizon of control. This feature together with the efficiency and performance of the parallel processing Kalman algorithms, combined with the present on board processing capabilities, guarantees the feasibility of real time, adaptive applications.

Notice that the proposed approach can be also be applied when an ODE mathematical model is not available. This can be done as long as dynamic system input output pairs are available to be used as training information, considering the structure of the numerical integrator with a feed forward network in place of the mean derivative function. Notice also that one could directly use an ODE numerical integrator as a dynamic system discrete model to play the role of an internal model in the predictive control scheme. However, in this case one would not have the possibility of adaptive control schemes, by exploring the learning capacity of the neural network and of on line updating its training.

The application of the proposed approach is not restricted to predictive control. It can be applied to any control scheme where an internal model of the controlled system is necessary.

Further studies shall evaluate the scheme adopted by Wang and Lin (1998), and depicted in Fig. 2. It is one where using the outputs of an Euler integrator the neural network is indirectly trained to learn the dynamic system mean derivative. In this paper it was only considered the scheme where the neural network is trained to directly learn the dynamic system mean derivative, which is then inserted in the structure of the Euler numerical integrator.

REFERENCES

Braun, M. (1983). *Differential equations and their applications: an introduction to applied mathematics*. 3rd ed., New York: Springer-Verlag, Applied Mathematical Sciences 15.

Carrara, V. (1997) *Redes Neurais Aplicadas ao Controle de Atitude de Satélites com Geometria Variável*. 202p. INPE-6384-TDI/603. Doctoral Thesis, Instituto Nacional de Pesquisas Espaciais-INPE, São José dos Campos, 1997.

Chandran, P. S. (1994). Comments on "comparative analysis of backpropagation and the extended kalman filter for training multilayer perceptrons". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 16, n. 8, pp. 862-863.

Chen, S., Billings, S. A., Luo, W. (1989). Orthogonal least squares methods and their application to nonlinear system identification. *Int. J. Control*, 50(5), pp. 1873-1896.

Chen, S., Billings, S. A., Cowan, C. F. N., Grant, P. M. (1990). Practical identification of NARMAX models using radial basis function. *Int. J. Control*, 52(6), pp. 1327-1350.

Chen, S.; Billings, S. A. (1992). Neural networks for nonlinear dynamic system modeling and identification. *Int. J. Control*, v. 56, n. 2, pp. 319-346.

Clarke, D.W., Mohtadi, C., Tuffs, P. S. (1987a). Generalized Predictive Control-Part I. The Basic Algorithm. *The Journal of IFAC the International Federation of Automatic Control. Automatica*, v. 23, n. 2, pp. 137-148.

Clarke, D.W., Mohtadi, C., Tuffs, P. S. (1987b). Generalized Predictive Control-Part II. Extensions and Interpretations. *The Journal of IFAC the International Federation of Automatic Control*. Automatica, v. 23, n. 2, pp. 149-160.

Cybenko, G. (1988). Continuous valued networks with two hidden layers are sufficient. *Technical Report*, Department of Computer Science, Tufts University.

Hornik, K.; Stinchcombe, M.; White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, v. 2, n. 5, pp. 359-366.

Hunt, K. J.; Sbarbaro, D.; Zbikowski, R.; Gawthrop, P. J. (1992). Neural networks for control systems – A survey. *Automatica*, v. 28, n. 6, pp. 1083-1112.

Kaplan, M. H.(1976). *Modern spacecraft dynamics & control*. New York: John Wiley & Sons.

Leontaritis, I. J., Billings, S. A. (1985a). Input-output parametric models for nonlinear system part I: Deterministic nonlinear systems. *Int. J. Control*, 41(2), pp. 303-328.

Leontaritis, I. J., Billings, S. A. (1985b). Input-output parametric models for nonlinear system part II: Stochastic nonlinear systems. *Int. J. Control*, 41(2), pp. 329-344.

Liu, G. P., Kadirkamanathan, V., Billings, S. A. (1998). Predictive Control for Nonlinear Systems Using Neural Networks. *Int. J. Control*, v. 71, n. 6, pp. 1119-1132.

Mills, P. M., Zomaya, A. Y., Tadé, M. O. (1994). Adaptive Model-Based Control Using Neural Networks. *Int. J. Control*, 60(6), pp. 1163-1192.

Munem, M. A., Foulis, D. J. (1978). *Calculus with Analytic Geometry*. Volumes I and II, Worth Publishers, Inc., New York.

Narendra, K. S.; Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, v. 1, n. 1, pp. 4-27.

Norgaard, M., Ravn, O., Poulsen, N. K., Hansen, L. K. (2000). Neural Networks for Modelling and Control of Dynamic Systems. Springer, London.

Rios Neto, A. (1997). Stochastic optimal linear parameter estimation and neural nets training in systems modeling. *RBCM – J. of the Braz. Soc. Mechanical Sciences*, v. XIX, n. 2, pp. 138-146.

Rios Neto, A. (2000). Design of a Kalman filtering based neural predictive control method. In: XIII CONGRESSO BRASILEIRO DE AUTOMÁTICA – CBA, 2000, UFSC (Universidade Federal de Santa Catarina), Florianópolis, Santa Catarina, Brazil, *CD-ROM Proceedings*, pp. 2130-2134.

Rios Neto, A. (2001). Dynamic systems numerical integrators control schemes. In: V CONGRESSO BRASILEIRO DE REDES NEURAIS, 2001, Rio de Janeiro, RJ, Brasil. *CD-ROM Proceedings*, pp. 85-88.

Sage A. P. (1968). *Optimum systems control*. Englewood Cliffs, NJ: Prentice-Hall, Inc..

Silva, J. A. (2001). *Controle preditivo utilizando redes neurais artificiais aplicado a veículos aeroespaciais.* 239 p. (INPE-8480-TDI/778). Doctoral Thesis, Instituto Nacional de Pesquisas Espaciais-INPE, São José dos Campos, Brazil.

Singhal, S., Wu, L. (1989). Training Multilayer Perceptrons with the Extended Kalman Algorithm. *In Advances in Neural Information Processing Systems*, VI, Morgan Kaufman Pub. Inc., pp 136-140.

Sokolnikoff, I. S.; Redheffer, R. M. (1966). *Mathematics of physics and modern engineering*. 2nd. ed., Tokyo: McGraw-Hill Kogakusha, LTD.

Tasinaffo, P. M. (2003). *Estruturas de integração neural feedforward testadas em problemas de controle preditivo*. 230 p. INPE-10475-TDI/945. Doctoral Thesis, Instituto Nacional de Pesquisas Espaciais-INPE, São José dos Campos, Brazil.

Tasinaffo, P.M., Rios Neto, A. (2003). Neural Numerical Integrators in Predictive Control tested in an Orbit Transfer Problem. In: CONGRESSO TEMÁTICO DE DINÂMICA, CONTROLE E APLICAÇÕES - DINCOM, 2, São José dos Campos: ITA, SP, Brasil, *CD-ROM Proceedings*, pp. 692-702.

Wang, Y.-J., Lin, C.-T. (1998). Runge-Kutta Neural Network for Identification of Dynamical Systems in High Accuracy. *IEEE Transactions On Neural Networks*, Vol. 9, No. 2, pp. 294-307, March.

Wertz, J. R. (1978). *Spacecraft attitude determination and control*. London: D. Reidel, Astrophysics and Space Science Library, v. 73.

Wilson, E. B. (1958). *Advanced Calculus*. New York: Dover Publications.

Zurada, J. M. (1992). *Introduction to Artificial Neural System*. St. Paul, MN, USA: West Pub. Co.

APPENDIX: RECURRENCE RELATIONS IN THE CALCULATION OF $H^U(T, I)$

Equations (18.a) to (18.d) can be solved only if the matrix $H^{u}(t,i)$ is known. For the case where discrete nonlinear dynamic system models using Euler numerical integrators with mean derivative functions are used, it can be calculated through the following equations (Tasinaffo, 2003):

$$\frac{\partial\, {}^{k+q}y^{i}}{\partial\, {}^{k}u} = \left(\frac{\partial\, \tan^{k+q-1}_{\alpha}{}^{i}}{\partial^{k+q-1}y^{i}} \cdot \Delta t + I \right) \cdot \frac{\partial^{k+q-1}y^{i}}{\partial^{k}u} \qquad (1A)$$

where, **q=2, 3, ..., $\mathbf{n}_{h.;}$** the notation is the same as in Section 2, and $\mathbf{n}_h$ is the number of steps in the finite horizon of optimization.

It is still necessary to calculate the back propagation relative only to the feed forward network, like is showed in figure 1A, to get the matrices (2A)

to (4A), in order to completely solve for $H^u(t,i)$.

$$\frac{\partial^{k+q-1} y^i}{\partial^{k} u} =$$

$$\Delta t \cdot \begin{bmatrix} \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha_1^i}{\partial^k u_1} & \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha^i}{\partial^k u_2} & \cdots & \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha_1^i}{\partial^k u_{n_{u'}}} \\ \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha_2^i}{\partial^k u_1} & \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha_2^i}{\partial^k u_2} & \cdots & \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha_2^i}{\partial^k u_{n_{u'}}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha_{n_{y'}}^i}{\partial^k u_1} & \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha_{n_{y'}}^i}{\partial^k u_2} & \cdots & \frac{\partial \tan_{\Delta t}^{k+q-2} \alpha_{n_{y'}}^i}{\partial^k u_{n_{u'}}} \end{bmatrix}_{n_{y'} \times n_{u'}}$$

(2A)

$$\frac{\partial \tan_{\Delta t}^{k+q-1} \alpha^i}{\partial^{k+q-1} y^i} =$$

$$\begin{bmatrix} \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_1^i}{\partial^{k+q-1} y_1^i} & \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_1^i}{\partial^{k+q-1} y_2^i} & \cdots & \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_1^i}{\partial^{k+q-1} y_{n_{y'}}^i} \\ \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_2^i}{\partial^{k+q-1} y_1^i} & \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_2^i}{\partial^{k+q-1} y_2^i} & \cdots & \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_2^i}{\partial^{k+q-1} y_{n_{y'}}^i} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_{n_{y'}}^i}{\partial^{k+q-1} y_1^i} & \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_{n_{y'}}^i}{\partial^{k+q-1} y_2^i} & \cdots & \frac{\partial \tan_{\Delta t}^{k+q-1} \alpha_{n_{y'}}^i}{\partial^{k+q-1} y_{n_{y'}}^i} \end{bmatrix}_{n_{y'} \times n_{y'}}$$

(3A)

$$\frac{\partial^{k+1} y^i}{\partial^{k} u} =$$

$$\Delta t \cdot \begin{bmatrix} \frac{\partial \tan_{\Delta t}^{k} \alpha_1^i}{\partial^k u_1} & \frac{\partial \tan_{\Delta t}^{k} \alpha_1^i}{\partial^k u_2} & \cdots & \frac{\partial \tan_{\Delta t}^{k} \alpha_1^i}{\partial^k u_{n_{u'}}} \\ \frac{\partial \tan_{\Delta t}^{k} \alpha_2^i}{\partial^k u_1} & \frac{\partial \tan_{\Delta t}^{k} \alpha_2^i}{\partial^k u_2} & \cdots & \frac{\partial \tan_{\Delta t}^{k} \alpha_2^i}{\partial^k u_{n_{u'}}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \tan_{\Delta t}^{k} \alpha_{n_{y'}}^i}{\partial^k u_1} & \frac{\partial \tan_{\Delta t}^{k} \alpha_{n_{y'}}^i}{\partial^k u_2} & \cdots & \frac{\partial \tan_{\Delta t}^{k} \alpha_{n_{y'}}^i}{\partial^k u_{n_{u'}}} \end{bmatrix}_{n_{y'} \times n_{u'}}$$

(4A)

In this case, the back propagation is given by (e.g., Carrara, 1997; Tasinaffo, 2003):

$$\frac{\partial y^l}{\partial \bar{y}^k} = \frac{\partial y^l}{\partial \bar{y}^{k+1}} \cdot W^{k+1} \cdot I_{f'(\bar{y}^k)} \text{ for } \mathbf{k=l-1, l-2} \qquad (5A)$$

where,

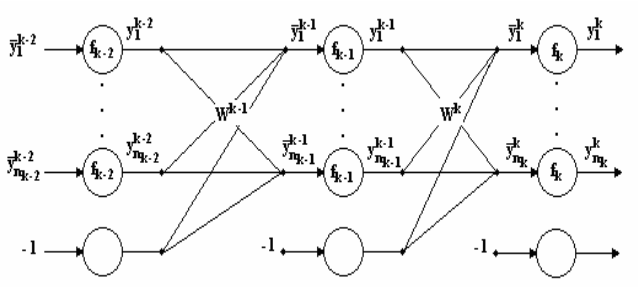

Figure 1A– The feed forward net.

$$\frac{\partial y^l}{\partial \bar{y}^l} = I_{f'(\bar{y}^l)} \qquad (6A)$$

$$I_{f'(\bar{y}^l)} = \begin{bmatrix} f'(\bar{y}_1^l) & 0 & 0 & 0 \\ 0 & f'(\bar{y}_2^l) & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & f'(\bar{y}_{n_k}^l) \end{bmatrix} \qquad (7A)$$

where the function $f'(.)$ is the derivate of the activation function of each neuron inside the feed forward net (Fig. 1A).

Equations 5A to 7A combined with equations (18.a) to (18.d) of Section 3 completely the problem of getting the matrix $H^u(t,i)$.