Learning and Nonlinear Models - Revista da Sociedade Brasileira de Redes Neurais (SBRN), Vol. 3, No. 2, pp. 98-109, 2005

MEAN DERIVATIVES BASED NEURAL EULER INTEGRATOR FOR NONLINEAR DYNAMIC SYSTEMS MODELING

Paulo M. Tasinaffo **Atair Rios Neto**
Instituto Nacional de Pesquisas Espaciais (INPE)
tasinafo@ita.br atairrn@uol.com.br

Abstract: The usual approach to nonlinear dynamic systems neural modeling has been that of training a feed forward neural network to represent a discrete nonlinear input-output NARMA (Nonlinear Auto Regressive Moving Average) type of model. In this paper, the recently developed alternative approach of combining feed forward neural networks with the structure of ordinary differential equations (ODE) numerical integrator algorithms is done in a way not yet considered. In this new approach, instead of using the neural network to learn the instantaneous derivative function of the ordinary differential equation (ODE) that describes the dynamic system, it is used to learn the dynamic system mean derivative function. This allows the use of an Euler structure to obtain a first order ODE neural integrator, which in principle can provide the same accuracy as that of any higher order integrator. The main objective is to have an approach in which the dynamic system neural modeling is simple. First in terms of the feed forward neural network training, since it has to learn only the algebraic and static functions of the system dynamic ODE mean derivatives. Second in terms of numerical complexity, since a first order integrator structure is sufficient to attain a specified accuracy. Test results of a practical problem, representing the dynamics of orbit transfer between the Earth and Mars, are used to illustrate the effectiveness of this new methodology.
Index Terms: Dynamic Systems Modeling, Neural Models, Numerical Integrators, Feed Forward Neural Nets.

1. INTRODUCTION

The capacity to represent nonlinear dynamic systems is a major driver behind the recent interest in artificial neural networks (ANN) (e.g. Narendra and Parthasarathy, 1990; Chen and Billings, 1992), opening new possibilities in terms of simulation, monitoring and control applications. Of special interest is the control application (e.g. Norgaard, 2000) where the neural network plays the role of an internal model in the control structure to represent the nonlinear plant response.

Because of their capacity to approximate any continuous function (Hornik et al, 1989; Cybenko, 1988), feed forward multi layer perceptrons (MLP) are used extensively as a NARMA (Nonlinear Auto Regressive Moving Average) model, in the representation of dynamic systems (e.g. Hunt et al, 1992; Chen and Billings, 1992).

Recent works have shown the possibility of inserting feed forward neural networks into the structure of an ODE integrator in order to get a discrete model representation of a given autonomous dynamic system, where the neural network needs only to learn the derivative function of the system. Wang and Lin (1998) applied this approach using a fourth order Runge-Kutta integrator to represent ordinary dynamic systems, and introduced in the literature the term *Runge-Kutta Neural Network.* Rios Neto (2001) independently explored the approach considering other ODE integrators structures, and showing its use in neural control. Since the dynamics is taken into account by the ODE numerical integrator algorithm, the neural network has to learn only an algebraic function that is named the changing rates of system states or the instantaneous derivative of ODE. Therefore, the direct benefit of this methodology is to lower the dimension of the neural network in terms of neurons and connections, facilitating its training and its implementation (Rios Neto and Tasinaffo, 2003).

In this work the possibility of using mean derivatives instead instantaneous derivative in the design of a neural numerical integrator is demonstrated and preliminarily tested. The resulting novel approach (Tasinaffo, 2003) further simplifies the structure and complexity of the ODE neural integrator representing discrete models of nonlinear dynamic systems. It is demonstrated that a neural integrator with a first order (Euler) structure is enough to get the same accuracy as that given by any higher order numerical integrator. Thus, the analytical expression for backpropagation in supervision training to neural Euler integrator with mean derivative is simpler than any another instantaneous derivative methodology, although the means derivative gets only fixed time step and instantaneous derivative is not limited by this.

In what follows, in Section 2, the possibility of representing discrete dynamic systems with Euler integrators using mean derivatives is given mathematical support. In Section 3, the proposed method is developed and analyzed. In Section 4, preliminary test results of application in a nonlinear dynamic system, corresponding to an Earth/Mars transfer orbit problem dynamics, are presented. And last, in Section 5, a few conclusions are drawn.

2. DISCRETE DYNAMIC SYSTEMS AND EULER INTEGRATOR WITH MEAN DERIVATIVES

Here, the mathematical fundamentals supporting the possibility of modeling discrete nonlinear dynamic systems by using mean derivative functions in the structure of Euler integrators are presented. These results show that, in principle, its is possible to have a discrete dynamic system model with accuracy equal to any higher order ODE integrator, using the structure of an Euler integrator, as long as sufficiently accurate values of the dynamic system mean derivatives are available. In the sequel the capacity of neural networks to represent nonlinear functions (e.g. Zurada, 1992) will be used to give a practical realization to this possibility.

Consider the following nonlinear autonomous system of ordinary differential equations, representing a given dynamic system:

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

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

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

It is convenient to introduce the following vector notation, to indicate possible initial condition sets at the initial time $\mathbf{t_o}$ and the respective trajectory solutions of (1a):

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

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

where, $\mathbf{i = 1, 2, ..., \infty;}$ and ∞ is adopted to indicate that the mesh of discrete initial conditions can have as many points as desired, starting from $\mathbf{y^i\,(t_o)}$ at initial time $\mathbf{t_o}$, that belong to a domain of interest $[\mathbf{y^{min}(t_o), y^{max}(t_o)}]$, where $\mathbf{y_j^{min}(t_o)}$ and $\mathbf{y_j^{max}(t_o)}$, **j = 1, 2, …, n,** are finite.

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

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

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

Consider the situation where $\dot{\mathbf{y}}^\mathbf{i} = \mathbf{f(y^i)}$ does not have an analytical solution, and this solution is to be represented by a set of discrete points of $\mathbf{y^i = y^i(t)}$, $[\mathbf{y^i(t + k \cdot \Delta t)} \;\; \mathbf{y^i[t + (k+1) \cdot \Delta t]} \; \ldots \;] \equiv [{}^k\mathbf{y^i} \;\; {}^{k+1}\mathbf{y^i} \; \ldots \;]$, for a given $\mathbf{\Delta t}$. To prepare for the possibility of getting these discrete points by using mean derivative functions in the structure of Euler integrators, the secant formed by the two points ${}^k\mathbf{y_j^i}$ and ${}^{k+1}\mathbf{y_j^i}$, belonging to the curve $\mathbf{y_j^i(t)}$, is defined as the straight-line segment joining them, and the correspondent tangent is:

$$\mathbf{tan_{\Delta t}}\,{}^k\alpha_j^i = \frac{{}^{k+1}\mathbf{y_j^i} - {}^k\mathbf{y_j^i}}{\mathbf{\Delta t}}, \;\; \mathbf{j = 1, 2, ..., n} \tag{3}$$

where, ${}^k\alpha_\mathbf{j}^\mathbf{i}$ is thus the angle of the secant which links the two points ${}^k\mathbf{y_j^i}$ and ${}^{k+1}\mathbf{y_j^i}$ belonging to the curve $\mathbf{y_j^i(t)}$.

Property 2 (P2) - *If* ${}^k\mathbf{y^i}$ *is a discrete solution of* $\dot{\mathbf{y}}^\mathbf{i} = \mathbf{f(y^i)}$ *and* $\mathbf{\Delta t \neq 0}$ *then* $\mathbf{tan_{\Delta t}}\,{}^k\alpha^\mathbf{i}$, *the vector of* $\mathbf{tan_{\Delta t}}\,{}^k\alpha_\mathbf{j}^\mathbf{i}$, **j = 1, 2, ..., n,** *exists and is unique.*

Two other important theorems, which relate the values of $\tan_{\Delta t}{}^{k}\alpha^{i}$ and $\tan_{\Delta t}{}^{k}\dot{\alpha}^{i}$, with the values of the mean derivatives calculated from $[\,{}^{k}y^{i} \;\; {}^{k+1}y^{i}\,]$ and $[\,{}^{k}\dot{y}^{i} \;\; {}^{k+1}\dot{y}^{i}\,]$, respectively, are the differential and integral mean value theorems (e.g., Wilson, 1958; Munem et al, 1978; Sokolnikoff et al, 1966), enunciated without demonstration in what follows.

Theorem 2 (T2) - *(Differential mean value theorem): If a function* $y_j^i(t)$*, 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 exists at least one* t_k^**,* $t_k < t_k^* < t_{k+1}$ *such that*

$$\dot{y}_j^i(t_k^*) = \frac{{}^{k+1}y_j^i - {}^{k}y_j^i}{\Delta t} \tag{4}$$

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

$$y_j(t_k^x)\cdot\Delta t = \int_{t_k}^{t_{k+1}} y_j{}^{i}(t)\cdot dt \tag{5}$$

In general t_k^* and t_k^x 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* ${}^{k+1}y^i$ *of the solution of the system of nonlinear differential equations* $\dot{y}^i = f(y^i)$ *can be determined through the relation* ${}^{k+1}y^i = \tan_{\Delta t}{}^{k}\alpha^{i}\cdot\Delta t + {}^{k}y^i$ *for a given* ${}^{k}y^i$ *and* Δt.

Demonstration: if $\dot{y}^i = \dfrac{dy^i}{dt} = f(y^i)$, then $\displaystyle\int_{{}^{k}y^i}^{{}^{k+1}y^i} dy^i = \int_{t_k}^{t_{k+1}} f(y^i)\cdot dt$. Consequently,

$${}^{k+1}y^i = \int_{t_k}^{t_{k+1}} f(y^i)\cdot dt + {}^{k}y^i \tag{6}$$

The application of the theorem of the mean integral value, **T3,** to the curve $\dot{y}^i(t)$ in the interval $[t_k, t_{k+1}]$ implies the existence of at least one t_k^x interior to $[t_k, t_{k+1}]$ such that

$$\dot{y}(t_k^x)\cdot\Delta t = \int_{t_k}^{t_{k+1}} \dot{y}^i(t)\cdot dt = \int_{t_k}^{t_{k+1}} f(y^i)\cdot dt \tag{7}$$

From the application of $\dot{y}^i(t_k^x) = \tan_{\Delta t}{}^{k}\alpha^{i}$, substituting $\dot{y}^i(t_k^x)$ in Eq. (7) and this in Eq. (6), it follows that:

$${}^{k+1}y^i = \tan_{\Delta t}{}^{k}\alpha^{i}\cdot\Delta t + {}^{k}y^i \tag{8}$$

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

$${}^{k+m}y_j^i = \sum_{l=0}^{m-1} \tan_{\Delta t}{}^{k+l}\alpha^{i}\cdot\Delta t + {}^{k}y_j^i \tag{9}$$

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

In the situation where the system of Eq. (1a) is autonomous, $y^{i_1}(t_1) = y^{i_2}(t_2)$, for $i_1 \neq i_2$ and $t_1 \neq t_2$, implies that $\dot{y}^{i_1}(t_1) = \dot{y}^{i_2}(t_2)$. This property establishes that two trajectories of $\dot{y} = f(y)$ starting from two different initial conditions, $y^{i_1}(t_o)$ and $y^{i_2}(t_o)$, for $i_1 \neq i_2$, will have the same derivatives only if $y^{i_1}(t_1) = y^{i_2}(t_2)$, even when $t_1 \neq t_2$; that is the system is autonomous. The question remaining is if the mean derivative $\tan_{\Delta t}{}^k\alpha^i$ is also autonomous, that is, time invariant. The properties that follow answer this question.

Property 4 (P4) – *If* $y^{i_1}(t)$ *and* $y^{i_2}(t)$ *are solutions of* $\dot{y} = f(y)$ *starting from* $y^{i_1}(t_o = 0)$ *and* $y^{i_2}(t_o = 0)$*, respectively, and if* $y^{i_1}(t_o = 0) = y^{i_2}(T)$ *for* ***T****>0, then* $y^{i_1}(\Delta t) = y^{i_2}(T + \Delta t)$ *for any given* Δt.

Demonstration: if $y^{i_2}(t)$ is a trajectory solution of $\dot{y} = f(y)$, then $y^{i_2}(t+T)$ belongs to this trajectory (see **P1**). By hypothesis, $y^{i_1}(0) = y^{i_2}(T)$ and thus $y^{i_1}(t) = y^{i_2}(t+T)$ for $t = 0$. Since **T1** guarantees uniqueness of solution, $y^{i_1}(t) = y^{i_2}(t+T)$ for every **t**, and, in particular, for $t = \Delta t$, it follows that $y^{i_1}(\Delta t) = y^{i_2}(\Delta t + T)$.

Property 5 (P5) - *If* $y^{i_1}(t_1) = y^{i_2}(t_2)$*, for* $i_1 \neq i_2$ *and* $t_1 \neq t_2$*, then,* $\tan_{\Delta t}\alpha^{i_1}(t_1) = \tan_{\Delta t}\alpha^{i_2}(t_2)$ *for* $\Delta t > 0$*, that is,* $\tan_{\Delta t}{}^k\alpha^{i_1}$ *is invariant in time.*

Demonstration: by definition, $\tan_{\Delta t}\alpha^{i_1}(t_1) = \frac{y^{i_1}(t_1+\Delta t) - y^{i_1}(t_1)}{\Delta t}$ and $\tan_{\Delta t}\alpha^{i_2}(t_2) = \frac{y^{i_2}(t_2+\Delta t) - y^{i_2}(t_2)}{\Delta t}$. Since $y_1^{i_1}(t_1) = y_2^{i_2}(t_2)$, by hypothesis, then, $y_1^{i_1}(t_1+\Delta t) = y_2^{i_2}(t_2+\Delta t)$, and $\tan_{\Delta t}\alpha^{i_1}(t_1) = \tan_{\Delta t}\alpha^{i_2}(t_2)$.

This result is very useful, since it determines that it is enough to know the values of $\tan_{\Delta t}{}^k\alpha^i$, for $i = 1, 2, ..., \infty$ at t_o, in a region of interest $[y_j^{min}, y_j^{max}]^n$, **j=1, 2, ..., n**, because for $t > t_o$ they will repeat. Figure 1 illustrates this property of autonomous dynamic systems, depicting that for any two different trajectories ($i_1 \neq i_2$) and at different instants of time ($t_1 \neq t_2$) the correspondent values of $\tan_{\Delta t}{}^k\alpha^i$ depend only on the values of the state, that is, whenever $y_1^{i_1}(t_1) = y_2^{i_2}(t_2)$, results $\tan_{\Delta t}\alpha^{i_1}(t_1) = \tan_{\Delta t}\alpha^{i_2}(t_2)$.

Notice also that due to the existence and uniqueness of $\tan_{\Delta t}{}^k\alpha^i$ the forward propagation of the dynamic system will have unique values of ${}^k\alpha(i)$ varying only in the interval $-\frac{\pi}{2} < {}^k\alpha(i) < \frac{\pi}{2}$. When the system is 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* ${}^k u$ *in each* $[t_k, t_{k+1}]$ *are used to solve the dynamic system:*

$$\dot{y}^i = f(y^i, u) \tag{10}$$

Demonstration: in this case the continuous function, $f(y^i, u)$, with ${}^k u$ approximated as a constant in $[t_k, t_{k+1}]$, can be viewed as being parameterized with respect to the control variable and, therefore, for any discrete interval the existence of the mean derivative $\dot{y}^i(t_k^*) = \frac{{}^{k+1}y^i - {}^k y^i}{\Delta t} = \tan_{\Delta t}{}^k\alpha^i$ is guaranteed and the result in Eq. (8) is still valid.

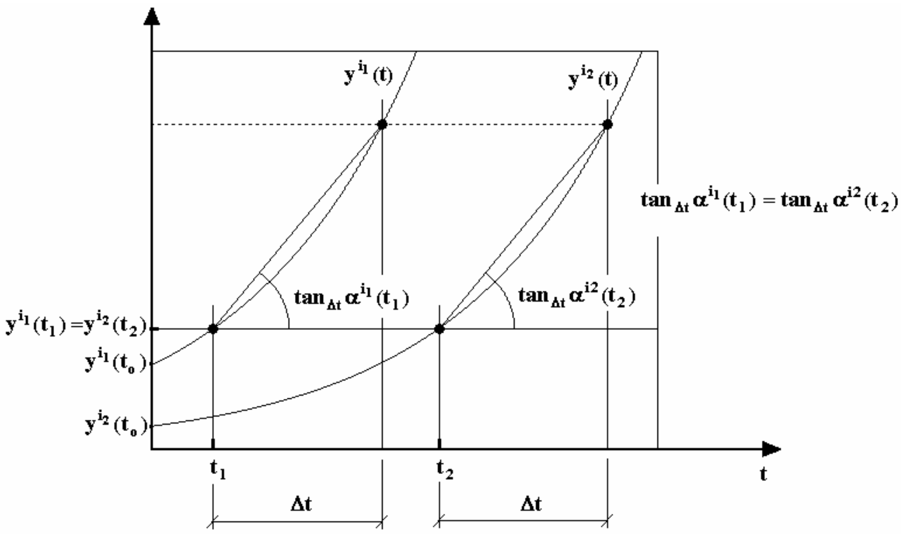

Figure 1 – Autonomous and time invariant $\mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\alpha^{\mathbf{i}}$.

3. NEURAL NUMERICAL INTEGRATORS WITH MEAN DERIVATIVES TO REPRESENT DYNAMIC SYSTEMS

In the previous section it was demonstrated that $\mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\alpha^{\mathbf{i}}$ has the following relevant characteristics:

a) In the interval $[\mathbf{t_k}, \mathbf{t_{k+1}}]$, $\mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\alpha^{\mathbf{i}} = \dfrac{{}^{\mathbf{k+1}}\mathbf{y^i} - {}^{\mathbf{k}}\mathbf{y^i}}{\Delta \mathbf{t}}$ is the mean derivative of $\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f(y^i, u)}$;

b) From **T1**, **P1** and **P2** it follows that the values of $\mathbf{tan}_{\Delta t}{}^{\mathbf{k+l}}\alpha^{\mathbf{i}}$ for $\mathbf{l = 0, 1, ..., (L\text{-}1)}$ exist and are unique, therefore $\mathbf{tan}_{\Delta t}{}^{\mathbf{k+l}}\alpha^{\mathbf{i}}$ is a static function with the same qualitative properties of the function of derivatives ${}^{\mathbf{k+l}}\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f}({}^{\mathbf{k+l}}\mathbf{y^i}, {}^{\mathbf{k+l}}\mathbf{u})$. As a matter of fact, it is true that $\lim\limits_{\Delta t \to 0} \mathbf{tan}_{\Delta t}{}^{\mathbf{k+l}}\mathbf{y^i} = \mathbf{f}({}^{\mathbf{k+l}}\mathbf{y^i}, {}^{\mathbf{k+l}}\mathbf{u})$. It is also important to observe that the function of derivatives $\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f(y^i, u)}$ does not depend on $\Delta \mathbf{t}$, but $\mathbf{tan}_{\Delta t}{}^{\mathbf{k+l}}\alpha^{\mathbf{i}}$ does. This property implies that the integration method with mean derivatives works with a fixed step size, while those with instantaneous derivatives allow the possibility of working with variable step sizes;

c) From **T2**, $\mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\alpha^{\mathbf{i}}$ is the exact derivative of at least one interior point of $[\mathbf{t_k}, \mathbf{t_{k+1}}]$, and this is another form of guaranteeing the existence of $\mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\alpha^{\mathbf{i}}$;

d) From **T3, T4** and **P3** it follows that the recurrent relation relating ${}^{\mathbf{k+1}}\mathbf{y^i}$, ${}^{\mathbf{k}}\mathbf{y^i}$ and ${}^{\mathbf{k}}\mathbf{u}$, to get a discrete solution of $\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f(y^i, u)}$, is given by ${}^{\mathbf{k+1}}\mathbf{y^i} = \mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\alpha^{\mathbf{i}} \cdot \Delta \mathbf{t} + {}^{\mathbf{k}}\mathbf{y^i}$, which is a simple Euler integration structure;

e) From **T5,** $\mathbf{f}({}^{\mathbf{k}}\mathbf{y^i}, {}^{\mathbf{k}}\mathbf{u})$ and $\mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\alpha^{\mathbf{i}}$ are both invariant in time, but parameterized with respect to ${}^{\mathbf{k}}\mathbf{u}$.

From the previous characteristics, one concludes that it is possible to use a neural network to represent a dynamic system with the Euler integration structure for a given step size $\Delta \mathbf{t}$. It is enough to consider the capacity of the neural network to approximate nonlinear functions (e.g., Zurada, 1992), using it to approximate the function correspondent to the dynamic system mean derivative.

The analysis of the Euler integration local error takes into consideration the exact value ${}^{\mathbf{k+1}}\mathbf{y^i}$ and the neural integrator approximated value ${}^{\mathbf{k+1}}\hat{\mathbf{y}}^{\mathbf{i}}$, as given by Eqs. (11a) and (11b).

$$ {}^{\mathbf{k+1}}\mathbf{y^i} = \mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\alpha^{\mathbf{i}} \cdot \Delta \mathbf{t} + {}^{\mathbf{k}}\mathbf{y^i} \tag{11a} $$

$$ {}^{\mathbf{k+1}}\hat{\mathbf{y}}^{\mathbf{i}} \cong (\mathbf{tan}_{\Delta t}{}^{\mathbf{k}}\boldsymbol{\alpha}^{\mathbf{i}} + \mathbf{e_m}) \cdot \Delta \mathbf{t} + {}^{\mathbf{k}}\mathbf{y^i} \tag{11.b} $$

where $\mathbf{e_m}$ is the output error of the neural network trained to learn the function of mean derivatives $\mathbf{tan_{\Delta t}}{}^{\mathbf{k}}\alpha^{\mathbf{i}}$ inside the domain of interest. Due to the capacity of approximation of the neural network, this error can be less than any specified value.

Accordingly, ${}^{\mathbf{k+1}}\hat{\mathbf{y}}^{\mathbf{i}}$, in Eq. (11b), can have the desired accuracy, given that for a fixed $\Delta\mathbf{t} > \mathbf{0}$, $\mathbf{e_m}$ can be made as small as specified, when the neural network is approximating the time invariant function of mean derivatives $\mathbf{tan_{\Delta t}}{}^{\mathbf{k}}\alpha^{\mathbf{i}}$ inside a domain of interest.

In the proposed approach, a first possibility was adopted as illustrated in Fig. 2, 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 ${}^{\mathbf{k}}\mathbf{y}^{\mathbf{i}}$ and control ${}^{\mathbf{k}}\mathbf{u}$, with a previously fixed discrete interval $\Delta\mathbf{t}$ The value of the training output pattern $\mathbf{tan_{\Delta t}}{}^{\mathbf{k}}\alpha^{\mathbf{i}} = \frac{{}^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}} - {}^{\mathbf{k}}\mathbf{y}^{\mathbf{i}}}{\Delta\mathbf{t}}$ is generated with the help of a numerical integrator of high order used to simulate one step ahead with negligible errors ${}^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$, the solution of the system $\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f(y^i, u)}$.

A second possibility that could be used, based on that adopted by Wang and Lin (1998), and depicted in Fig. 3, 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, ${}^{\mathbf{k+1}}\hat{\mathbf{y}}^{\mathbf{i}}$, the value of state estimated by the neural Euler numerical integrator, is the output value compared to the training pattern ${}^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$ to generate the error signal for the supervised training.

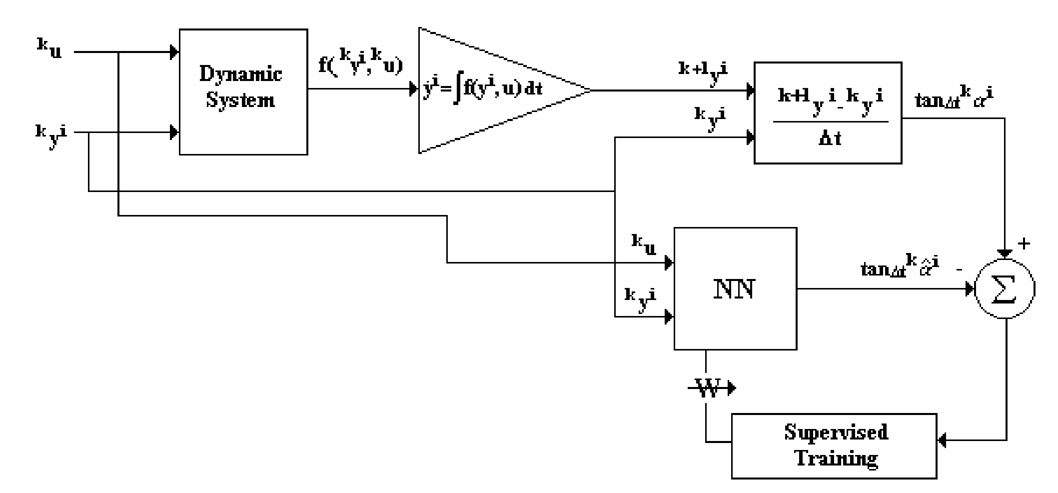

Figure 2 – Supervised Training of Mean Derivatives of $\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f(y^i, u)}$.

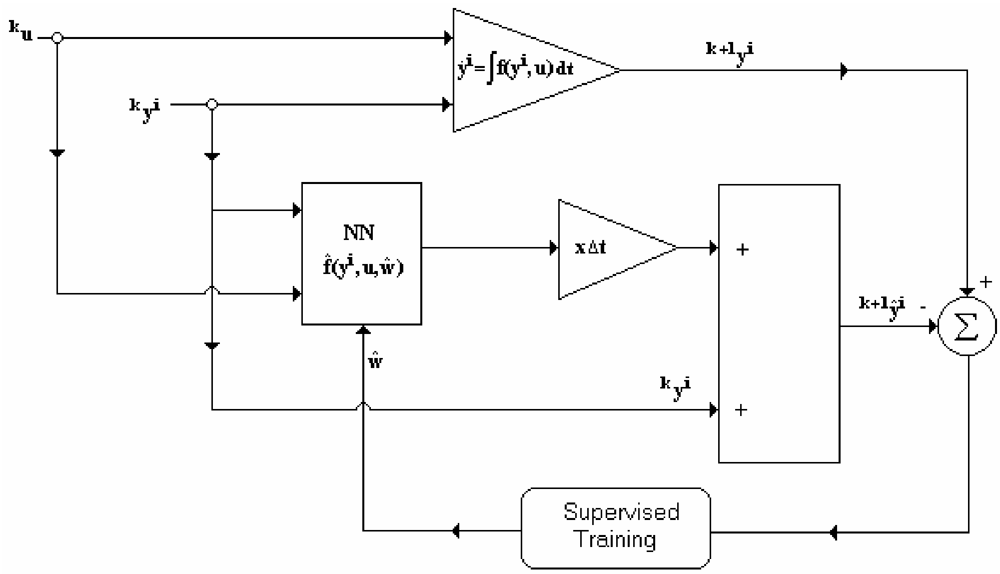

Figure 3 – Supervised Training of an Euler Neural Integrator Using Mean Derivatives of $\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f}(\mathbf{y}^{\mathbf{i}}, \mathbf{u})$.

The relation of recurrence between the true values of $^{\mathbf{k}}\mathbf{y}^{\mathbf{i}}$ and $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$ is expressed by $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}} = \mathbf{tan}_{\Delta t}\,^{\mathbf{k}}\alpha^{\mathbf{i}} \cdot \Delta\mathbf{t} + {}^{\mathbf{k}}\mathbf{y}^{\mathbf{i}}$. Therefore, if $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$ is obtained with negligible errors, either from the use of a high order integrator or experimentally, then $\mathbf{tan}_{\Delta t}\,^{\mathbf{k}}\alpha^{\mathbf{i}} \cong \hat{\mathbf{f}}(^{\mathbf{k}}\mathbf{y}^{\mathbf{i}}, {}^{\mathbf{k}}\mathbf{u}, \hat{\mathbf{w}})$, which is the mean derivative approximated by the neural network. It should be observed that if $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$ is obtained as an approximation from a Euler integration, the neural network will approximate the function of derivatives $\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f}(\mathbf{y}^{\mathbf{i}}, \mathbf{u})$. Figure 4 illustrates this situation. Consider $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$, $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}_{\mathbf{a}}$ and $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}_{\mathbf{e}}$, representing the exact value of the solution of $\dot{\mathbf{y}}^{\mathbf{i}} = \mathbf{f}(\mathbf{y}^{\mathbf{i}}, \mathbf{u})$ at $\mathbf{t}_{\mathbf{k+1}}$, the approximate value of $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$ obtained from a high order numerical integrator, and the approximate value of $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$ obtained from an Euler integrator, respectively. As indicated by Fig. 4, when it is assumed that $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}} = {}^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}_{\mathbf{e}}$, in the scheme of Fig. 3, then it follows that $\mathbf{tan}\,^{\mathbf{k}}\theta^{\mathbf{i}} = \mathbf{f}(^{\mathbf{k}}\mathbf{y}^{\mathbf{i}}, {}^{\mathbf{k}}\mathbf{u})$. On the other hand, taking $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}} = {}^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}_{\mathbf{a}}$, in this scheme, and if $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}_{\mathbf{e}}$ is away from $^{\mathbf{k+1}}\mathbf{y}^{\mathbf{i}}$, then, during the training phase, the neural network will converge to $\mathbf{tan}_{\Delta t}\,^{\mathbf{k}}\alpha^{\mathbf{i}}$, the function of mean derivatives, instead of converging to $\mathbf{f}(\mathbf{y}^{\mathbf{i}}, \mathbf{u})$.

The use of this approach to nonlinear dynamic systems neural modeling shall provide the following advantageous characteristics: (i) less complexity in terms of architecture (number of layers and neurons), and less difficulty in terms of training, as compared to a NARMA type model, since the discrete modeling of the dynamic system is accomplished using feed forward neural networks in the structure of ODE numerical integrators, and the neural network has to only learn an algebraic and static function, instead of having to learn the dynamic system discrete model based in delayed inputs, as is the case with a NARMA type of model; (ii) the accuracy of any high order neural integrator with the minimum possible complexity, since the neural network is inserted in a simple first order Euler structure; and (iii) in the case of a control application, it permits an internal dynamic system model that simplifies the determination of partial derivatives, without destroying the accuracy in the approximation of dynamic system outputs.

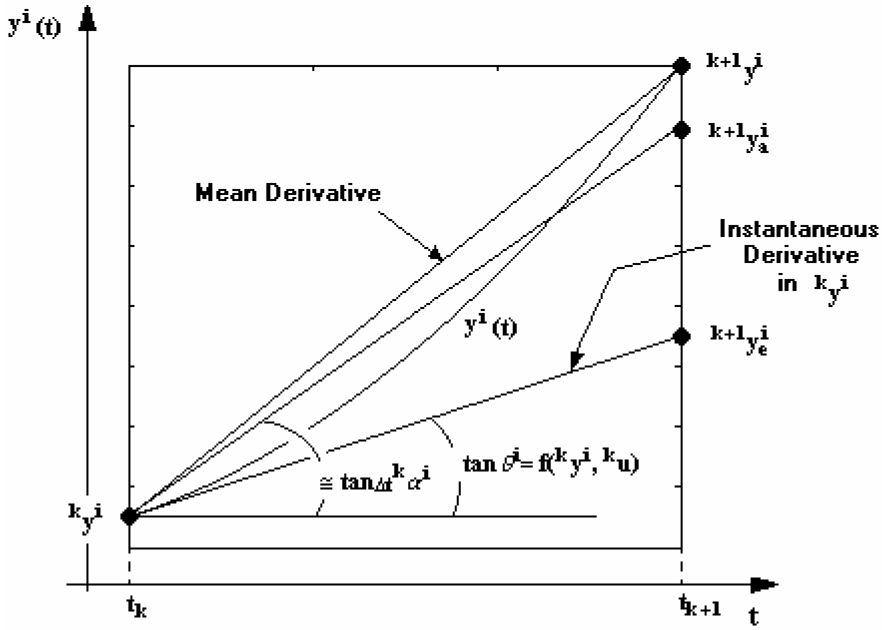

Figure 4 – Representation of mean $\tan_{\Delta t}{}^k\alpha^i$ and instantaneous $\tan {}^k\theta^i = f({}^k y^i, {}^k u)$ derivatives, for $\dot{y}^i = f(y^i, u)$.

4 DEMONSTRATION TEST AND RESULTS

This new approach was tested in a practical problem of ODE modeling for the dynamics of an orbit transfer between Earth and Mars. In this problem the state variables are the rocket mass **m**, the orbit radius **r**, the radial speed **u** and the transversal speed **v**, and the control variable is the thrust steering angle θ, measured from local horizontal (e.g., Sage, 1968):

$$\dot{m} = -0.0749 \tag{12a}$$

$$\dot{r} = u \tag{12b}$$

$$\dot{u} = \frac{v^2}{r} - \frac{\mu}{r^2} + \frac{T \cdot \sin\theta}{m} \tag{12c}$$

$$\dot{v} = \frac{-u \cdot v}{r} + \frac{T \cdot \cos\theta}{m} \tag{12d}$$

where the variables have been normalized with: $\mu = 1.0$, the gravitational constant; **T**=*0.1405*, the thrust; $t_o=0$ and $t_f=5$, initial and final times. The normalized unit of time is taken to be *58.2* days; and the unit of distance is equal to the astronomical unit (mean distance from Earth to Sun).

In all the MLP training, in the tests that follow, the same Extended Kalman filtering algorithm, with the same calibration conditions was used, as implemented by Tasinaffo (2003) based on a version from the literature, as presented in Rios Neto (1997). This algorithm results from viewing the training of a fedforward neural network as a problem of parameter estimation where the neural network can be treated as a parameterized mapping formally represented as

$$\hat{y}(t) = \hat{f}(x(t), w) \tag{13}$$

where w is the vector of weight parameters to be identified by fitting a given data set of input-output training patterns

$$\{(x(t), y(t) \quad t = 1, 2, ..., L\} \tag{14}$$

An iterative algorithm is obtained when a linear approximation is taken, and the following stochastic linear estimation problem is solved with a sequential Kalman filtering in a ith iteration (Rios Neto, 1997):

$$\overline{w} = w(i) + \overline{e} \tag{15a}$$

$$\mathbf{z(t,i) = H(t,i) \cdot w(i) + v(t)} \tag{15b}$$

$$\mathbf{E[\bar{e}] = 0, \qquad E[ee^T] = \bar{P}} \tag{15c}$$

$$\mathbf{E[v(t)] = 0, \quad E[v(t)\, v^T(t)] = R(t)} \tag{15d}$$

$$\mathbf{z(t,i) \equiv \alpha(i) \cdot \left[y(t) - \bar{y}(t,i)\right] + \hat{f}_w(x(t),\bar{w}(i)) \cdot \bar{w}(i)} \tag{15e}$$

$$\mathbf{H(t,i) \equiv \hat{f}_w(x(t),\bar{w}(i))} \tag{15f}$$

where, $\mathbf{i = 1, 2, ..., I;}$ $\mathbf{t= 1,2,\dots, L}$; $\mathbf{\bar{w}(i)}$ is the a priori estimate of **w** coming from the previous iteration, starting with $\mathbf{\bar{w}(1) = \bar{w}}$; $\mathbf{\bar{y}(t,i) = \hat{f}(x(t),\bar{w}(i))}$ is the output of the feedforward network; $\mathbf{\hat{f}_w(x(t),\bar{w}(i))}$ is the matrix of first partial derivatives with respect to **w**; $\mathbf{0 < \alpha(i) < 1}$ is a parameter to be adjusted in order to guarantee the hypothesis of linear perturbation. In the MLPs training, the calibration conditions were as following: initial weights were randomly generated from Gaussian distributions with unit standard deviation; $\mathbf{\bar{P}}$ was taken to be the identity matrix at the beginning of each new iteration; initially **R**=0.1**I** and after each iteration was taken as this initial value multiplied by the square root of attained mean square error; and the parameter to guarantee linear perturbation was taken as $\mathbf{\alpha(i) = 0.1}$. With this calibration, each iteration took a few minutes (at most 8 minutes) in a PC computer (Athlon, 256 Mbytes, 1.6 GHz) and of the order of 1000 iterations were necessary.

Initially tests were conducted to evaluate the capacity of the mean derivatives based neural Euler integrator in generating an accurate discrete model for the dynamics of the test problem. A 4^{th} order Runge-Kutta integrator (e.g. Lambert, 1973) was used to generate the training and test patterns, from initial conditions and control values randomly generated from uniform distributions inside the hypercube defined by the domains of the variables as in Table 1. A MLP neural network with the following characteristics was empirically adjusted and used to learn the mean derivatives, according to the scheme of Fig. 2: Five inputs (four state variables and one control); one hidden layer with 41 biased neurons using hyperbolic tangent (λ=2) as activation function, and an output layer with 4 biased neurons, yielding the approximation of the mean derivatives vector. The MLP was trained with the Kalman filtering algorithm, until the learning stabilized with 3600 patterns for training and 1400 patterns for testing. Mean square errors attained by the procedure, for this specific problem, were $2.4789.10^{-6}$ and $2.7344.10^{-6}$, respectively. To assess the capacity of generalization, a test trajectory was generated with the 4^{th} order Runge Kutta, from standard initial conditions (**m** = 1.0, **r** = 1.0, **w** = **v** = 0) using an open loop random control law, with values of control in each discrete interval being the outcomes of a uniform distribution between $-\pi$ and $+\pi$. From the same initial conditions and with the same sequence of controls a correspondent approximation trajectory using the mean derivative Euler neural integrator was generated. The results are shown in Fig. 5. These results indicate the effectiveness of the proposed approach in providing accurate discrete models of dynamic systems.

For the sake of comparison, a neural network with the same architecture as the previous one was trained under similar conditions in terms of generation of training and test patterns, with the same Kalman filtering algorithm, to learn the instantaneous derivatives used in the structures of 4^{th} order Runge Kutta and Adams Bashfort numerical integrators. In this case, the learning stabilized with 9000 training patterns and 1000 testing patterns, when mean square errors of $3.3580.10^{-5}$ and $3.3565.10^{-5}$ were respectively attained. These errors were the best errors that could be achieved before the Kalman filter training saturated. From the same initial conditions and with the same sequence of controls used in the test trajectory, correspondent approximation trajectories using these 4^{th} order, instantaneous derivative based neural integrators were generated. In this case the results were not satisfactory, the 4^{th} order, instantaneous derivative based neural integrators were not accurate discrete models for the dynamics of the test problem and were not able to avoid bad results in terms of cumulative errors as depicted in Fig. 6, thus indicating that a more complex neural network, in terms of number of neurons in the hidden layer, might be necessary for the approximation of the instantaneous derivatives, in order to provide an accuracy of approximation as good as that provided by the mean derivatives based neural Euler integrator.

TABLE 1 – Hypercube of Training Domain of Variables

Variable	Max	Min.
m	**0.2**	**1.0**
R	**0.8**	**2.0**
W	**-1.5**	**+1.5**
V	**0.0**	**1.5**
θ	**-1.2.** π	**+1.2.** π

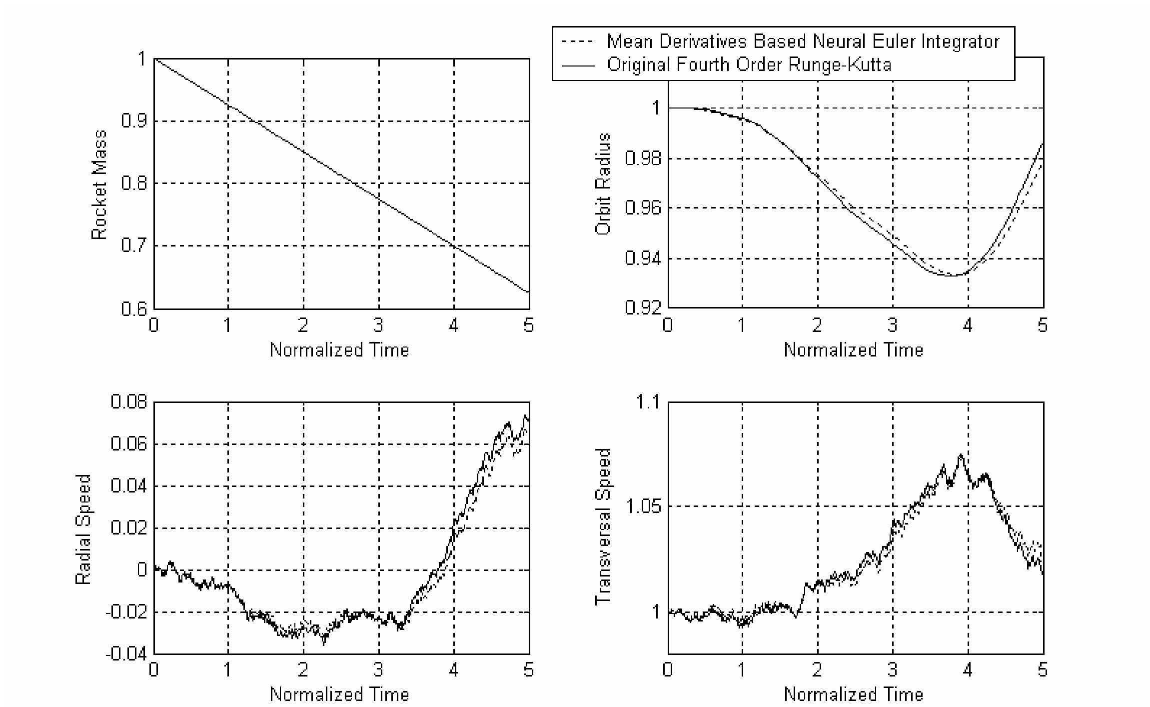

Figure 5 – Numerical Results in the Earth to Mars Orbit Transfer Dynamics with Randomly Applied Control (Mean Derivatives Based Neural Euler Integrator with Δ T=0.01).

Figure 7 presents the results when again a neural network with the same architecture as the one in the mean derivatives based neural Euler integrator is trained under similar conditions in terms of generation of training and test patterns, with the same Kalman filtering algorithm, but now playing the role of a zero order NARMA model, that is, having to directly learn the discrete model for the dynamics of the test problem, as represented by the 4th order Runge-Kutta integrator which is playing the role of the validation model. In this case, 3600 training patterns and 1400 testing patterns were used and mean square errors of $2.1115.10^{-7}$ and $2,1983.10^{-7}$ were respectively attained. Visual inspection is enough to conclude that the results in this case were the worst. As a matter of fact, this comparison is not fair, since this kind of results was expected when the neural network with the same architecture was used, because now the neural net needs to have a more complex architecture to afford the capacity of learning the dynamics of the problem.

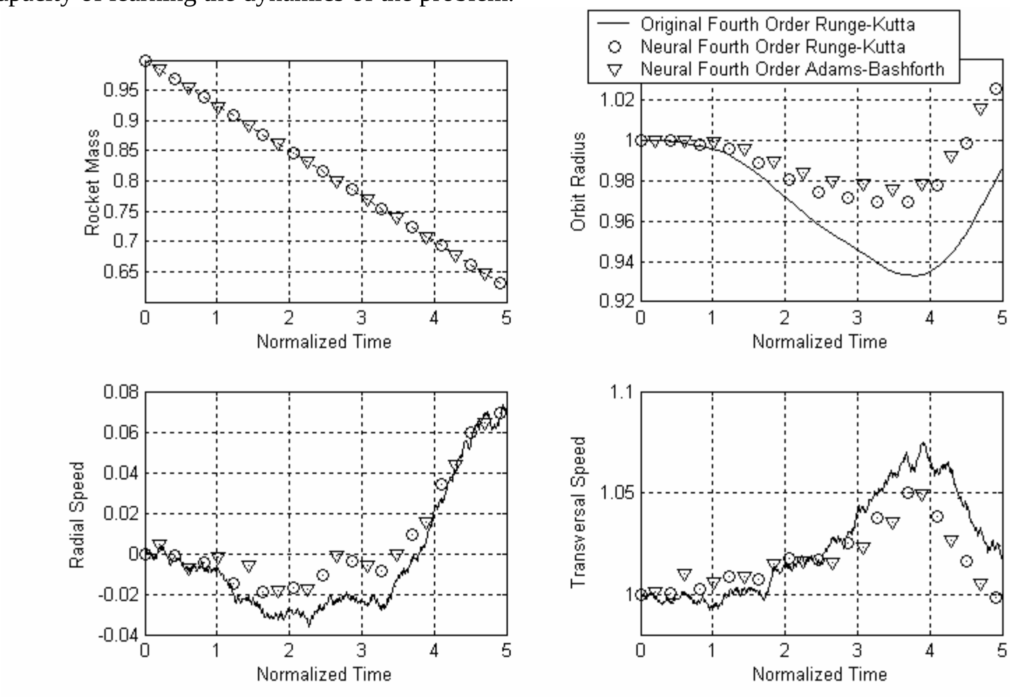

Figure 6 – Numerical Results in the Earth to Mars Orbit Transfer Dynamics with Randomly Applied Control (instantaneous derivative function with Δ t=0.01).

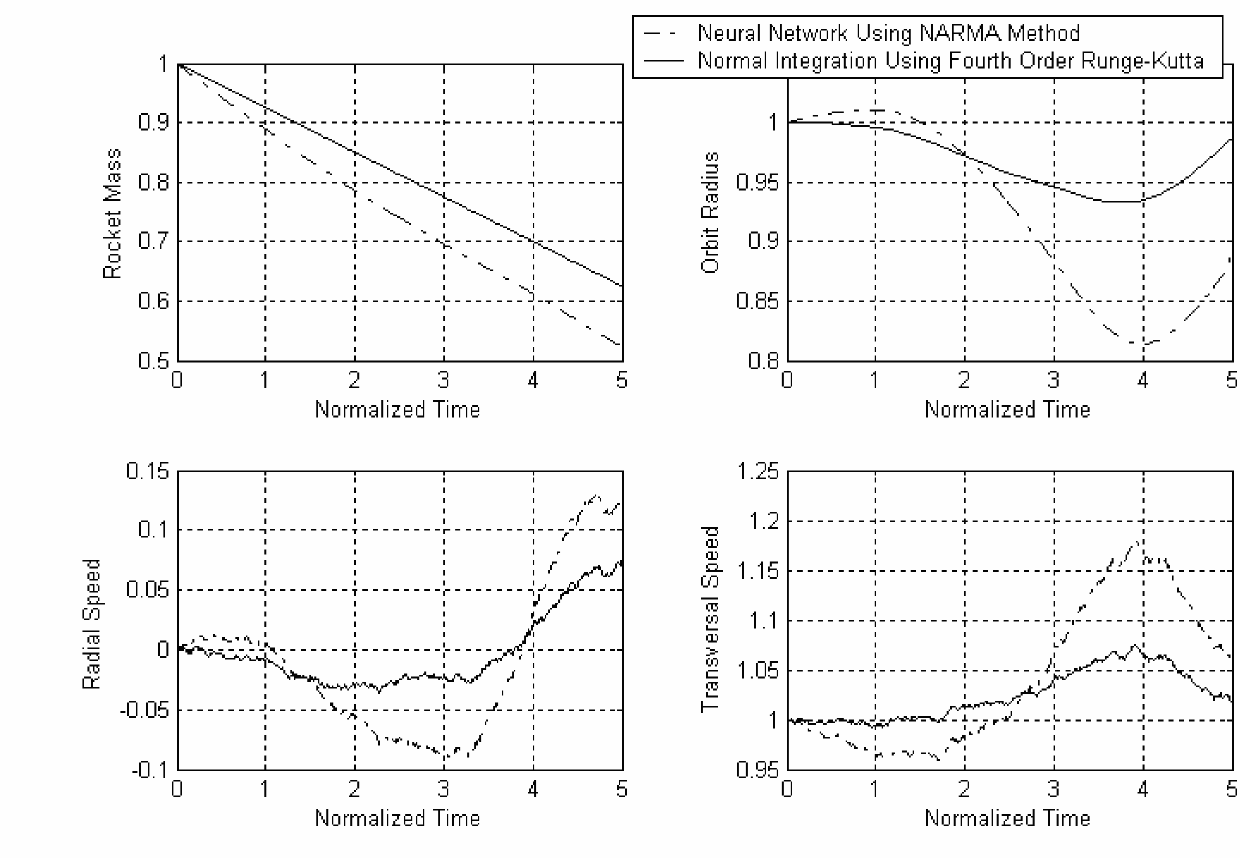

Figure 7 – Numerical Results in the Earth to Mars Orbit Transfer Dynamics with Randomly Applied Control (NARMA method with Δ T=0.01).

5 CONCLUSIONS

A new approach for using neural networks to model discrete dynamic systems was presented, mathematically demonstrated and preliminarily tested. By exploring the possibility of using ODE numerical integrators as discrete models of dynamic systems, it was demonstrated that the structure of these numerical integrators could be simplified to the limit of an Euler first order integrator without compromising the accuracy of approximation, when a neural model of the dynamic system mean derivatives is adopted. This led to an approach which not only offers the advantages of ODE neural numerical integrators as compared to neural networks used as a NARMA model, but also provides an additional advantage in terms of numerical complexity. In fact, with the structure of an Euler first order integrator it is possible to get the same accuracy as that given by any higher order numerical integrator.

These characteristics were verified in tests carried out with a representative example of a nonlinear dynamic system. The mean derivatives based neural Euler integrator was able to generalize and to provide approximations with the same level of accuracy of the Fourth Order Runge Kutta validation model used to simulate the true model. The use of neural networks playing the role of dynamic system instantaneous derivatives in ODE neural integrators of fourth order, or representing a discrete nonlinear zero order input-output NARMA type of model, with the same architecture and under the same training conditions as those used with the mean derivatives based neural Euler integrator, did not produce as good results, substantiating the advantage of the latter method in its ability to provide more accurate results with lower numerical cost and complexity.

6 REFERENCES

[01] – Braun, M. *Differential Equations and Their Applications*. Springer-Verlag, 3rd Edition, Applied Mathematical Sciences 15, New York, 1983.
[02] – Chen, S., Billings, S. A. Neural Networks for Nonlinear Dynamic System Modelling and Identification. *Int. J. Control*, Vol. 56, No. 2, pp. 319-346, 1992.
[03]- Cybenko, G. Continuous Valued Networks With Two Hidden Layers Are Sufficient. *Technical Report*, Department of Computer Science, Tufts University, 1988.
[04] – Hornik, K., Stinchcombe, M., White, H. Multilayer Feed forward Networks are Universal Approximators. *Neural Networks*, Vol. 2, No. 5, pp. 359-366, 1989.
[05] – Hunt, K. J., Sbarbaro, D., Zbikowski, R., Gawthrop, P. J. Neural Networks for Control Systems – A Survey. *Automatica*, Vol. 28, No. 6, pp. 1083-1112, 1992.

[06] – Lambert, J. D. *Computational methods in ordinary differential equations*. John Wiley & Sons, New York, 1973.
[07] – Munem, M. A., Foulis, D. J. *Calculus with Analytic Geometry*. Volumes I and II, Worth Publishers, Inc., New York, 1978.
[08] – Narendra, K. S., Parthasarathy, K. Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Transactions on Neural Networks*, Vol. 1, No. 1, pp. 4-27, march 1990,.
[09] – Norgaard, M., Ravn, O., Poulsen, N. K., Hansen, L. K. *Neural Networks for Modelling and Control of Dynamic Systems*. Springer, London, 2000.
[10] - Rios Neto, A. Stochastic Optimal Linear Parameter Estimation and Neural Nets Training in Systems Modeling. *RBCM-J. of the Braz. Soc. Mechanical Sciences*, XIX(2), pp. 138-146, 1997.
[11] - Rios Neto, A., Tasinaffo, P.M. Feed forward Neural Networks Combined With a Numerical Integrator Structure for Dynamic Systems Modeling. *Proceedings of the 17th International Congress of Mechanical Engineering-COBEM 2003*, November 2003, São Paulo, SP, Brazil.
[12] – Rios Neto, A. Dynamic Systems Numerical Integrators in Neural Control Schemes. *V Congresso Brasileiro de Redes Neurais*, April 2001, Rio de Janeiro, RJ, Brazil.
[13] – Sage A. P. *Optimum Systems Control*. Prentice-Hall, Inc., Englewood Cliffs, N. J., 1968.
[14] – Sokolnikoff, I. S., Redheffer, R. M. *Mathematics of Physics and Modern Engineering*. McGraw-Hill Kogakusha, LTD., Second Edition, 1966.
[15] - Tasinaffo, P. M. *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, 2003.
[16] – Wang, Y.J., Lin, C.T. 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 1998.
[17] – Wilson, E. B. *Advanced Calculus*. Dover Publications, New York, 1958.
[18] – Zurada, J. M. *Introduction to Artificial Neural System*. West Pub. Co., St. Paul, MN, USA, 1992.