Use of Neural Networks in Sounding Rocket Trajectory Reconstitution

Alexandre N. Barbosa[1] and José Demisio S. da Silva[2]
[1]Master's Program in Applied Computing - CAP
[2]Laboratory for Computing and Applied Mathematics - LAC
Brazilian National Institute for Space Research - INPE
C. Postal 515 – 12245-970 – São José dos Campos - SP
BRAZIL
E-mail: n.barbosa@iae.cta.br, demisio@lac.inpe.br

Abstract

The neural networks, which are a model of how the brain works, are well known as universal approximators. This article takes advantage of such characteristic on presenting an alternative way for reconstituting a sounding rocket trajectory, instead of using the conventional approach, which spends too much effort. The basic idea is concerned to the use of a neural network to map the reconstituted trajectory, solving drop-outs, which is obtained by a linear interpolation of the difference between nominal and actual flight.

Keywords: neural networks, trajectory reconstitution, sounding rocket.

1. INTRODUCTION

A rocket launch mission takes into account three main aspects. The first one is concerned to the payload experiments. The second one is the rocket which meets the mission requirements. The third one is the launch infra-structure that encompasses the systems to track the rocket, such as the telemetry and radar systems. In Brazil, the priority is the radar. There exist at least two radar systems, one for proximity and another one for accuracy. Both provide the instantaneous rocket position and velocity. They should carry out without halts, but tracking interruptions could occur due to atmospheric conditions, malfunctions and so on. Such problem happened with the first flight of the Brazilian rocket VSB-30, during its launch operation in Alcantara Launch Center (CLA), in October 2004. When the tracking vanishes simultaneously in both radars, they are guided by the information of the nominal trajectory, in order to retry the tracking.

After the flight, the post-flight analysis starts. The Flight Dynamics Subdivision of the Space and Aeronautics Institute in Brazil is responsible for studying the rocket trajectory and performing its post-flight analysis. In order to carry out such analysis, computational applications are employed. Among those applications, it is the Rocket Simulation >> ROSI << for trajectory calculation (Kramer, 1976). If the actual trajectory is not quite tracked by the radars, it is performed its reconstitution by using the ROSI and updating its input data with actual conditions.

This paper treats with a typical problem of signal loss, or drop-out. For the time that the radars have been blinded, the nominal characteristics of the flight are not realized in the actual trajectory. Therefore, the ROSI software is employed to reconstitute such characteristics. Although the current method to reconstitute the trajectory is probably the most accurate, it takes too much effort to updating the input data of the ROSI. Besides, the complexity of the reconstitution using the ROSI increases when burning and atmosphere phases are taken into account of the occluded stream of the actual trajectory (Louis, 2004).

2. THE PROPOSED METHOD

Searching by an alternative method, this paper introduces the use of neural networks, in a preliminary approach, in order to reconstitute the trajectory of a sounding rocket, instead of using the ROSI. The basic idea is to calculate the difference from both nominal and actual trajectories, and reconstitute the occluded nominal characteristics, that are not realized in the actual trajectory by using a linear interpolation on the difference

between them. After that, a multilayer Perceptron is trained to map the reconstituted actual trajectory.

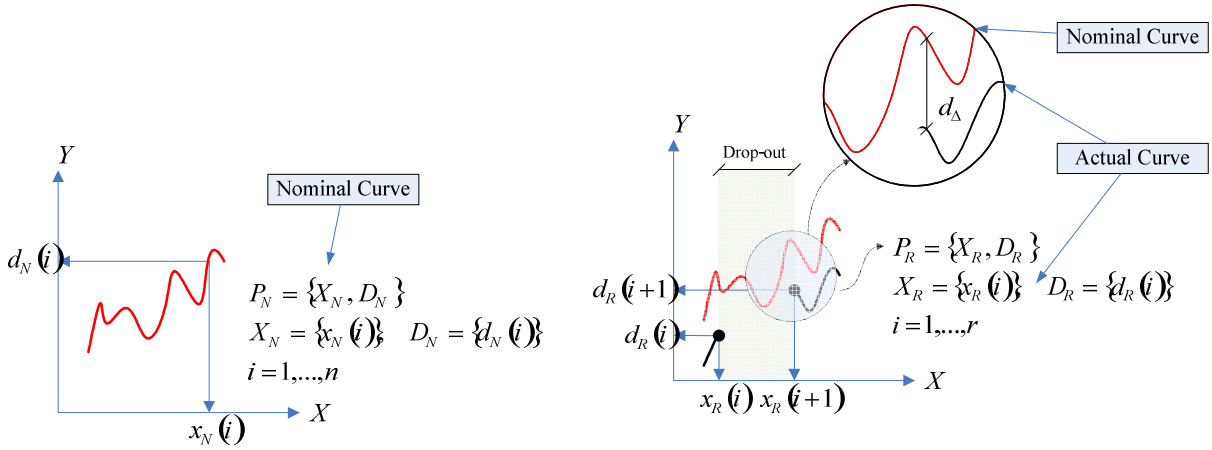


Figure 1 The Nominal and Actual Curves

The actual curve reconstitution is obtained by linear interpolation of the difference between the nominal and actual curves (Figure 1). After that, a multilayer Perceptron is defined to map the actual trajectory (Figure 2).

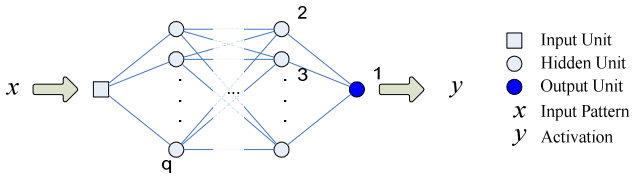


Figure 2 The Neural Network Definition

The neural units are identified sequentially, from 1 to q (Figure 2). In this manner, the neural network architecture is defined by changing the number of hidden layers and units per layer. The importance of testing different architectures is that the best one among them, which has the minimum quadratic error, can be chosen to map different curves with different sinuosities.

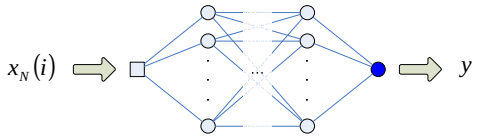
In order to minimize the quadratic error in the output of the unit 1 by adapting the synaptic weights of the multilayer Perceptron, the Backpropagation algorithm is employed for training the neural network. Such algorithm was developed by David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams, in 1986. The term is an abbreviation for “backwards propagation of errors” (Haykin, 1999) (Hecht-Nielsen, 1990). The error is calculated with respect to the reconstituted patterns of the actual trajectory.

The algorithm, which follows below, describes step by step the training of the multilayer Perceptron, taking into account the linear interpolation to estimate the actual curve.

Step 0: The steps from 1 to 10 are repeated until the quadratic error (Equation 2) start to converge asymptotically or the number of epochs be greater than what was previously defined.

Step 1: For each pattern $x_N(i)$, $i = 1,...,n$, the steps from 2 to 8 are performed.

Step 2: The nominal input $x_N(i)$ is presented to the neural network that gives the result of its activation y.



Step 3: The indices j^- and j^+ of the actual input data, which are closest from each other and bound $x_N(i)$, are found.

$x_R(j^-) \leq x_N(i) \leq x_R(j^+)$ and $x_R(j^-) \neq x_R(j^+)$

Both indices are initialized with value -1.

Step 4: The indices i^- and i^+ of the nominal input data, which are respectively closest from $x_R(j^-)$ and $x_R(j^+)$, are found.

$$\left.\begin{aligned} i^- &= \arg\min_k \left\{ \left\| x_R(j^-) - x_N(k) \right\| \right\} \\ i^+ &= \arg\min_k \left\{ \left\| x_R(j^+) - x_N(k) \right\| \right\} \end{aligned}\right\} \quad k = 1,\ldots,n$$

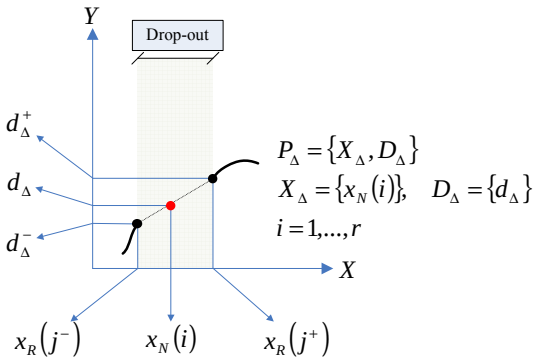
Step 5: Using the closest nominal input data, whose indices were found in step 4, the nominal output pattern for each one, $x_R(j^-)$ and $x_R(j^+)$, $j^{\pm} \neq -1$, is estimated and assigned to d_N^- and d_N^+.

$d_N^- = d_N(i^-)$ and $d_N^+ = d_N(i^+)$

Step 6: The difference from nominal and actual curve d_Δ is calculated, using linear interpolation.

P_Δ is the set of patterns that approximates the difference from nominal and actual curves.

If $j^{\pm} \neq -1$ is true, the difference d_Δ is calculated by a linear interpolation (Equation 1).

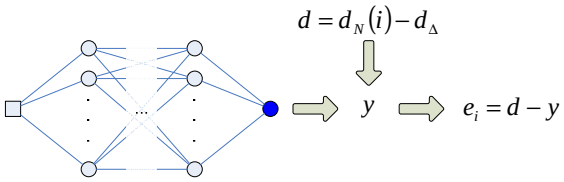


$$\left.\begin{aligned} d_\Delta^- &= d_N^- - d_R(j^-) \\ d_\Delta^+ &= d_N^+ - d_R(j^+) \end{aligned}\right\} \Rightarrow d_\Delta = \frac{\left(x_N(i) - x_R(j^-)\right) \cdot d_\Delta^+ + \left(x_R(j^+) - x_N(i)\right) \cdot d_\Delta^-}{x_R(j^+) - x_R(j^-)} \qquad (1)$$

In another hand, if $j^+ = -1 \vee j^- = -1$ is true, the value of d_Δ is equal to the closest actual input.

$$d_\Delta = \begin{cases} d_\Delta^+, & j^+ \neq -1 \wedge j^- = -1 \\ d_\Delta^-, & j^+ = -1 \wedge j^- \neq -1 \end{cases}$$

Step 7: The error in the output of the neural network is calculated.



Step 8: The Backpropagation algorithm is employed to adapt the synaptic weights of the neural network.

Step 9: The quadratic error is calculated.

$$E = \frac{1}{2 \cdot n} \cdot \sum_{k=1}^{n} e_k^2 \tag{2}$$

Step 10: The terminal condition is verified.

The algorithm above describes the proposed method. But, its implementation should take into account the computation efficiency. This way, the estimated output patterns for training the neural network should be obtained before performing such algorithm by finding and keeping the indices j^-, j^+, i^- and i^+.

3. CASE STUDIES

In order to demonstrate the proposed method and verify for what conditions it diverges, a drop-out is simulated on a quite covered flight, such as the second one of the VSB-30, in such way that it is possible to calculate the error with respect to the omitted part of the actual trajectory. In addition to this, an enhanced approach is briefly introduced for further studies.

As a first case study, a single curve was taken from the trajectory of the VSB-30, which is relevant at all for flight analysis, the total velocity of the rocket as a function of time.

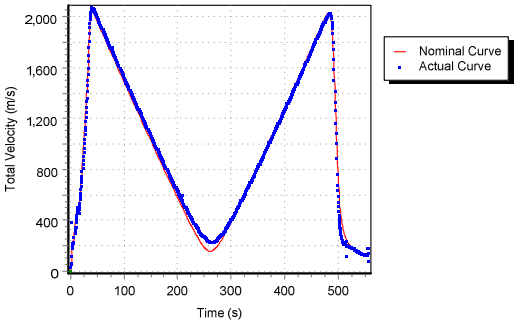
The second flight of the VSB-30 has spent 550 seconds. In the current case study, a drop-out on the total velocity actual data was provoked from 30 to 300 seconds of the flight.

The neural network training was performed with success. It was used a multilayer Perceptron with three hidden layers and eight units per layer. 351 patterns were presented during the training and 350 for testing.

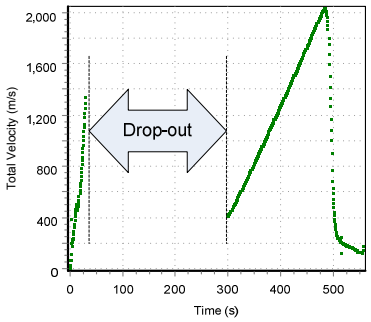
Figure 3 VSB-30

The relative error, which was achieved after training, was about 8% (Figure 4). It is figured that the ROSI has better results if compared with the proposed method. It would be pretty good to compare both methods, but it still remains as a suggestion for further studies.

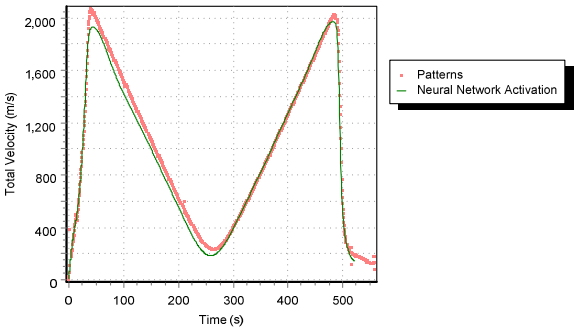
Graphic 1: The Nominal and Actual Total Velocity



Graphic 2: The Actual Curve with a Drop-out



Graphic 3: The Reconstituted Actual Curve



Graphic 4: The Neural Network Performance

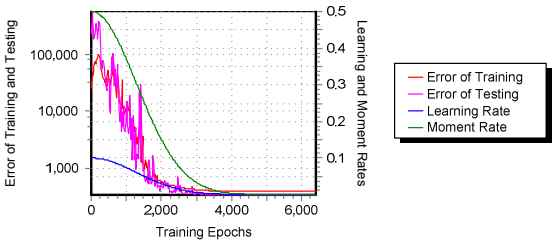
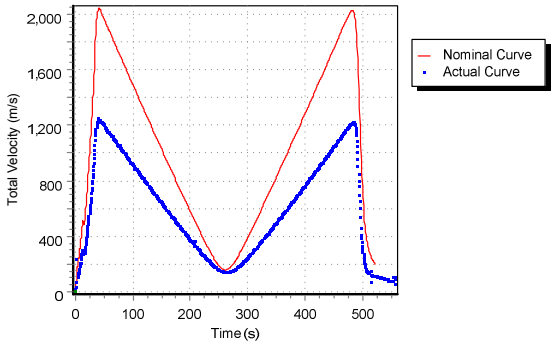


Figure 4 The Reconstitution of the Total Velocity

In a second case study, the actual curve was multiplied by a number in the interval (0, 1) and the same drop-out was provoked so that the transformed curve could be understood as an anomaly of the flight.



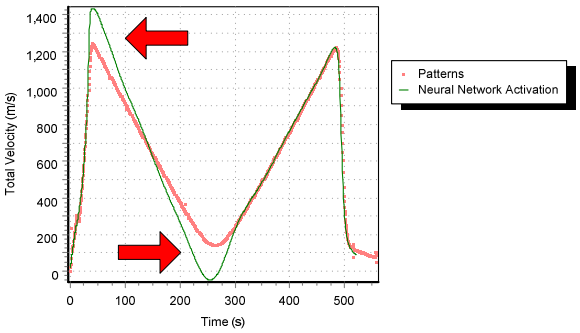


Figure 5 The Reconstitution of an Anomalous Trajectory

The method diverges obviously if the difference between the nominal and actual curves is nonlinear (Figure 5). Such situation could occur if the actual trajectory is anomalous. In this case, the proposed method should not be used.

Looking forward to enhancing the method, one could try another approach. Instead of using a linear interpolation to estimate de difference between the nominal and actual trajectories, an additional neural network could be wanted to perform it. Such approach is probably more suitable for reconstituting a slightly anomalous trajectory.

4. CONCLUSION

The proposed method could be an alternative way for reconstituting the rocket trajectory. It takes advantage of its simplicity when compared with the ROSI software. In addition, it could be extended to others categories of problems that are quite similar to the problem of occlusion.

5. REFERENCES

1. HAYKIN, S. (1999), *Neural Networks: A Comprehensive Foundation. 2. Ed*. Prentice Hall.

2. HECHT-NIELSEN, R (1990), *Neurocomputing*. Addison-Wesley.

3. KRAMER, H.J.; CRAUBNER, A.; ZIEGLTRUM, W. (1976) *ROSI Rocket Simulation*. DFVLR TN 12/76.

4. LOUIS, João E. (2004), *Reconstituição da trajetória do veículo VSB-30 V01*. ASE-RT-021-2004.