Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 6, No. 2, May 2015), Pages: 73-82
www.jacr.iausari.ac.ir

# Accurate Prediction of DGPS Correction using Neural Network Trained by Imperialistic Competition Algorithm

**Mohammad Reza Mosavi** ✉[1], **Hodeis Nabavi** [2]

*1) Department of Electrical Engineering, Iran University of Science and Technology, Narmak, Tehran 16846-13114, Iran, Corresponding Author, Professor*

*2) Department of Computer and Electrical Engineering, Islamic Azad University of Babol, Babol, Iran*

m_mosavi@iust.ac.ir; hodeis_nabavi@yahoo.com

**Received: 2014/05/27; Accepted: 2014/07/09**

**Abstract**

*This paper presents an accurate Differential Global Positioning System (DGPS) using multi-layered Neural Networks (NNs) based on the Back Propagation (BP) and Imperialistic Competition Algorithm (ICA) in order to predict the DGPS corrections for accurate positioning. Simulation results allowed us to optimize the NN performance in term of residual mean square error. We compare results obtained by the NN technique with BP and ICA. Results show a good improvement obtained by the application of the NN trained by the ICA. The experimental results on measurement data demonstrate that the prediction total RMS error using NN trained by the ICA learning algorithm are 0.8273 and 0.7143 m, before and after selective availability, respectively.*

***Keywords:*** *DGPS Corrections, NNs, BP, ICA*

## 1. Introduction

The Global Positioning System (GPS) comprises a constellation of continuously orbiting satellites run by the United States Department of Defense (DOD). Two levels of positioning accuracy are provided, Standard Positioning Service (SPS) using the Coarse/Acquisition (C/A) code signals and Precise Positioning Service (PPS). PPS is encoded and not available to civilian users. SPS was deliberately degraded to provide an accuracy of 10 m (95% probability). This error source was removed after 1 May 2000 [1]. Table 1 shows the source of GPS position errors after intentional error of Selective Availability (SA) was turned off.

***Table 1. The source of GPS position errors after SA***

| Source of error | Amount of error (meters) |
|---|---|
| Ionosphere | 4 |
| Clock | 2.1 |
| Ephemeris | 2.1 |
| Troposphere | 0.7 |
| Receiver noise | 0.5 |
| Multi-path | 1 |
| Total | 10.4 |

GPS accuracy can be improved over this 10 m figure through the use of Differential GPS (DGPS), where a reference station broadcasts corrections on common view satellites on a regular basis to the remote GPS receiver, which provides a corrected position output. However, any interruption of DGPS service will cause loss of navigation guidance, which might cause vehicle accident, particularly during of precision approach and landing [2].

Service interruptions can be the result of a short term loss of lock of the GPS signals at the DGPS ground base station, or they can be unintentional interrupts of DGPS correction generation caused by hardware or software failures. Such short term performance degradations can be overcome by applying predictions of the DGPS corrections. From the nature of the DGPS corrections, the Neural Network (NN) modelling is a suitable way to precisely predict the DGPS corrections for a limited period [3].

This paper thus focuses on the continuity performance of the DGPS corrections. Differential correction techniques are used to enhance the quality of location data gathered using GPS receivers. Differential correction can be applied in real-time directly in the field or when post processing data in the office. Although both methods are based on the same underlying principles, each accesses to different data sources and achieves different levels of accuracy. Combining both methods provides flexibility during data collection and improves data integrity. From the nature of the DGPS corrections, NN modelling with different learning algorithms such as Back-Propagation (BP) provides a suitable way to precisely predict the DGPS corrections for a limited period [4].

The BP learning rule is a central rule in much current works in learning in artificial NN. BP provides a computationally efficient method of changing the weights in a feed-forward network, with differentiable activation function units, to learn a training set of input-output examples. BP is a gradient-descent search algorithm. Learning by BP is normally slow due to the characteristics of the error surface on which the weights are being navigated. Different kinds of enhancements have been proposed for BP in order to speed up the convergence up, and to avoid local minima in the error surface. Because of the gradient descent nature of BP learning, it is very sensitive to weight initialization. The convergence speed of BP is dependent on the learning rate. Many alternatives have been proposed to make the learning rate adaptive [5].

Imperialistic Competition Algorithm (ICA) is an algorithm for optimization inspired by the imperialistic competition. Like other evolutionary ones, the ICA algorithm starts with an initial population. Population individuals called country are in two types: colonies and imperialists that all together form some empires. Imperialistic competition among these empires forms the basis of the proposed evolutionary algorithm. During this competition, weak empires collapse and powerful ones take the possession of their colonies. Imperialistic competition hopefully converges to a state in which there exist only one empire and its colonies are in the same position and have the same cost as the imperialist. Applying the ICA algorithm to some of benchmark cost functions, shows its ability in dealing with different types of optimization problems [6].

The concern of this paper is designing is a multi-layered NN based on the BP and ICA for DGPS corrections prediction. Proposed methods validity is checked with different experiments on collected real data. This paper is prepared as follows. Section 2 focuses on the proposed methods for DGPS corrections prediction using multi-layered NN.

Section 3 discusses the experimental results. Discussions are reported with measured data. Section 4 provides the conclusion.

## 2. DGPS Corrections Prediction with Multi-Layered Neural Network

Multi-layer NNs are feed-forward NN models which are also referred to as Multi-Layer Perceptrons (MLPs). The addition of a hidden layer of neurons in the perceptron allows the solution of non-linear problems. The complexity of the MLP network can be changed by varying the number of layers and the number of units in each layer. Given enough hidden units and data, it has been shown that MLPs can approximate virtually any function to any desired accuracy [7].

In this paper, NNs refer to the prediction of DGPS corrections. They convey DGPS corrections at time $n+N$, from the $p$ time steps back from time $n$. To decrease the error of DGPS, an analysis of two learning algorithms on MLP NN, including BP and ICA, has been proposed.

### *2.1. NN Learning based on the BP*

BP model is the most popular model in the supervised learning architecture because of the weight error correct rules. It is considered a generalization of the delta rule for non-linear activation functions and multi-layer networks [8].

In a BP NN, the learning algorithm has two phases. First, a training input pattern is presented to the network input layer. The network propagates the input pattern from a layer to another layer until the output pattern is generated by the output layer. If this pattern is different from the desired output, error is calculated and then propagated backward through the network from the output layer to the input layer. The weights are modified as the error is propagated.

The BP training algorithm is an iterative gradient designed to minimize the mean square error between the actual output of multi-layer feed-forward perceptron and the desired output. It requires the continuous differentiable non-linearity. The following assumes a sigmoid logistic non-linearity [9]:

**Step1.** Initialize weights and offsets: Set all weights and node offsets to small random values.

**Step2.** Present inputs and desired outputs: Present a continuous valued inputs vector $X_0, X_1, \dots, X_{N-1}$ and specify the desired outputs $d_0, d_1, \dots, d_{M-1}$.

**Step3.** Calculate actual output: Use the sigmoid non-linearity from above and formulas to calculate outputs $Y_0, Y_1, \dots, Y_{M-1}$.

**Step4.** Adapt weights: Use a recursive algorithm starting at the output nodes and working back to the first hidden layer. Adjust weights by:

$$\mathrm{w_{ij}}(\mathrm{t}+1) = w_{ij}(t) + n\delta_j x_i' \quad (1)$$

In this equation, $w_{ij}(t)$ is the weight from hidden node $i$ or from an input to node j at time t, $w_j'$ is either the output of node $i$ or is an input, η is a gain term and $\delta_j$ is an error term for node j. If node j is an output node, then:

$$\delta_j = y_j(1 - y_j)(d_j - y_j) \quad (2)$$

Where $d_j$ is the desired output of node j and $y_j$ is the actual output. If node j is an internal hidden node, then:

$$\delta_j = x'_j(1 - x'_j)\sum_k \delta_j^m w_{jk} \quad (3)$$

Where k is over all nodes in the layers above node j. Internal node thresholds are adapted in a similar manner by assuming they are connection weights on links from auxiliary constant-valued inputs. Convergence is sometimes faster if, a momentum term is added and weight changes are smoothed by:

$$w_{ij}(t+1) = w_{ij}(t) + n\delta_j x'_i + \alpha(w_{ij}(t) - w_{ij}(t-1)) \;\; ; \;\; 0 < \alpha < 1 \quad (4)$$

**Step5.** Repeat by going to step2.

BP is applicable just for networks with differentiable activation functions. The speed of convergence and the possibility of ending up in a local minimum of the error function are other typical problems about BP algorithm.

***2.2. NN Learning based on ICA***

The goal of optimization is to find an optimal solution in terms of the variables of the problem formed an array of variable values to be optimized which the term "country" is used for this array. In a $N_{Var}$ – dimensional problem, a country is an $1 \times N_{Var}$ array. This array is defined by:

$$Country = [p_1, p_2, p_3, ..., p_{N_{var}}] \quad (5)$$

The variable values in the country are represented as floating point numbers. The cost of a country is found by evaluating the cost function f at the variables $(p_1, p_2, p_3, ..., p_{N_{Var}})$ then:

$$Cost = f(Country) = f(p_1, p_2, p_3, ..., p_{N_{var}}) \quad (6)$$

To form the initial empires, the colonies should be divided among imperialists based on their power. That is the initial number of colonies of an empire should be directly proportionate to its power. To divide the colonies among imperialists proportionally, we define the normalized cost of an imperialist by:

$$C_n = c_n - \max\{c_i\} \quad (7)$$

Where $c_n$ is the cost of n-th imperialist and $C_n$ is i-th normalized cost. Having the normalized cost of all imperialists, the normalized power of each imperialist is defined by:

$$p_n = \frac{C_n}{\sum_{i=1}^{N_{imp}} C_i} \quad (8)$$

Total power of an empire is mainly affected by the power of imperialist country. But the power of the colonies of an empire has an effect, albeit negligible, on the total power of that empire. We have modeled this fact by defining the total cost by:

$$T.C_n = Cost(imperialist_n) + \varepsilon mean\{Cost(colonies \;\; of \;\; empire_n)\} \quad (9)$$

Where, $T.C_n$ is the total cost of the n-th empire and $\varepsilon$ is a positive number which is considered to be less than 1. The procedure for ICA is summarized as follows [6]:

1. Initialize the empires.
2. Move the colonies toward their relevant imperialist.
3. If there is a colony in an empire which has lower cost than that of imperialist go to 4 otherwise 5.
4. Exchange the positions of that imperialist and the colony.
5. Compute the total cost of an empire.
6. Pick the weakest colony from the weakest empire and give it to the empire that has the most likelihood to process it.
7. If there is a empire with no colonies go to 8 otherwise.
8. Eliminate this empire.
9. If stop condition satisfied go to 10 otherwise 2.
10. Stop.

## 3. Experimental Results

The measurement data sets in Iran University of Science and Technology were used for evaluation of the performance of the proposed NNs and the optimal selection of these method parameters was based on the experimental results. The proposed NNs performance depends on increasing of the NNs order and the improvement of training patterns which increases the memory for software implementation and also the structure complexity for hardware implementation. Thus, there must be a trade-off in selecting the order of NNs and training patterns between CPU time and accuracy of methods. The most appropriate structure for the proposed NNs is $(3,5,1)$, i.e. three inputs in input layer, five neurons in hidden layer and one neuron in output layer, here.

All input and output variables are normalized in the range [0, 1] for preparing the training data in order to reduce the training time. Observation at time $n$ is applied to NNs input. The value of instant must be predicted by the network in time $n+1$. For the prediction $y(n) = x(n+1)$. After training the NNs, their performance on the measured data set had been used for their assessment. In order to compare conveniently, we use Root Mean Square (RMS) to measure the performance of our NNs as:

$$RMS = \sqrt{\frac{1}{M}\sum_{i=1}^{i=M}(d_i - y_i)^2} \quad (11)$$

Where, $y_i$ is the network output, $d_i$ is the target value from the test data set and $M$ is the number of tests. Figures 1 to 4 show $d_x$, $d_y$ and $d_z$ predictions for 1000 test data by using proposed NNs trained by BP and ICA, before and after SA, respectively. $d_x$, $d_y$ and $d_z$ are errors of position components. They are obtained difference between the measurement and accurate values, i.e.: $dx, dy, dz_{Error} = dx, dy, dz_{Measurement} - dx, dy, dz_{Accurate}$. The obtained results have been reported in Tables 2 and 3.

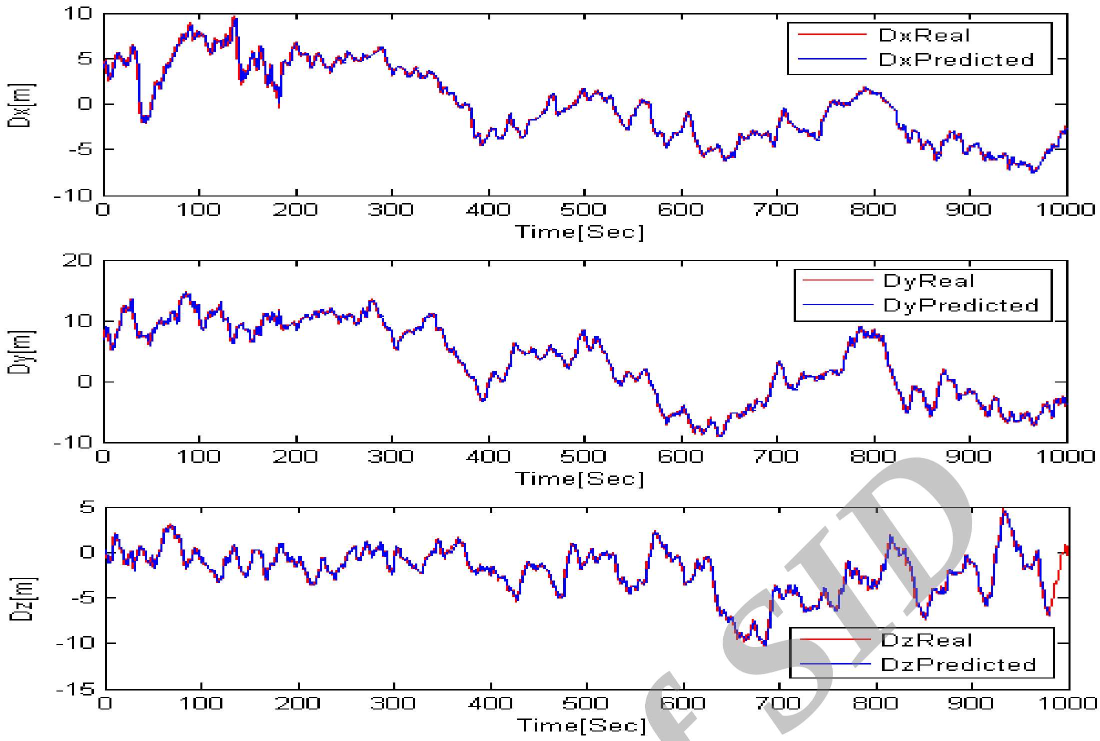


***Figure 1. Using MLP NN trained by BP learning algorithm for prediction of DGPS corrections (SA on)***

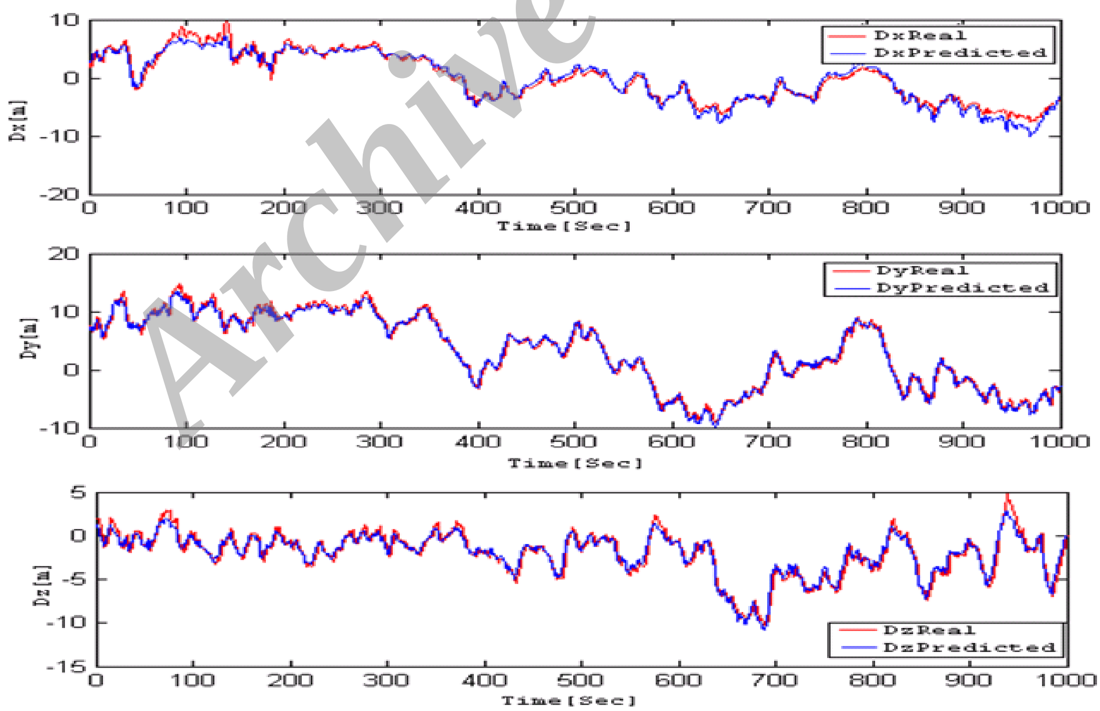


***Figure 2. Using MLP NN trained by ICA learning algorithm for prediction of DGPS corrections (SA on)***

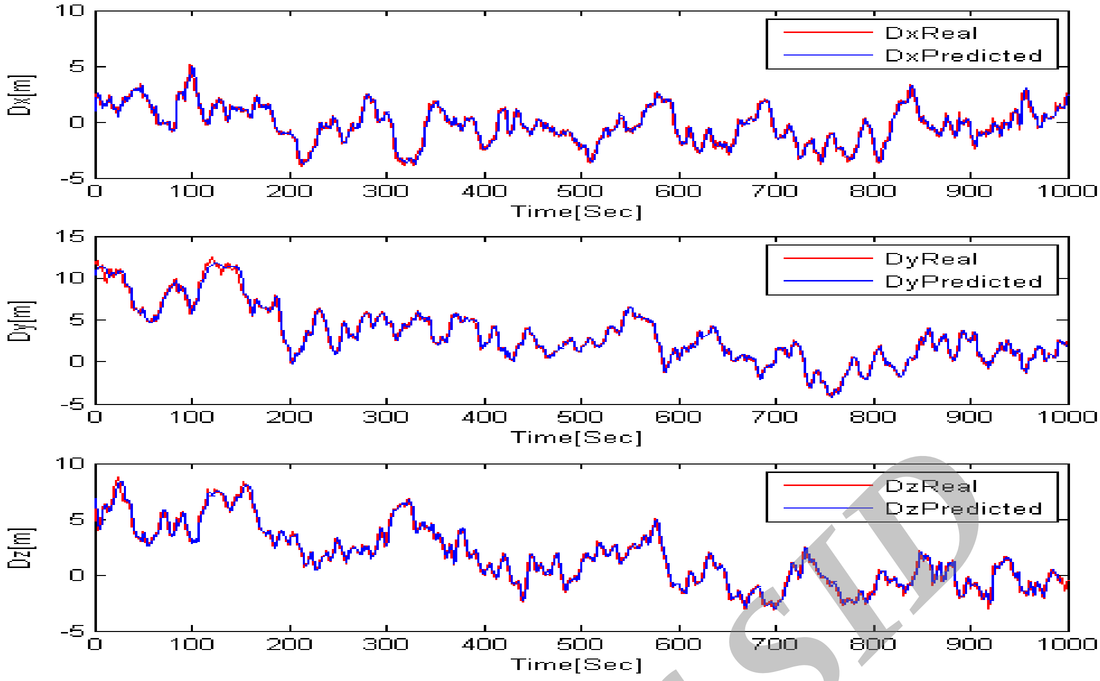


***Figure 3. Using MLP NN trained by BP learning algorithm for prediction of DGPS corrections (SA off)***

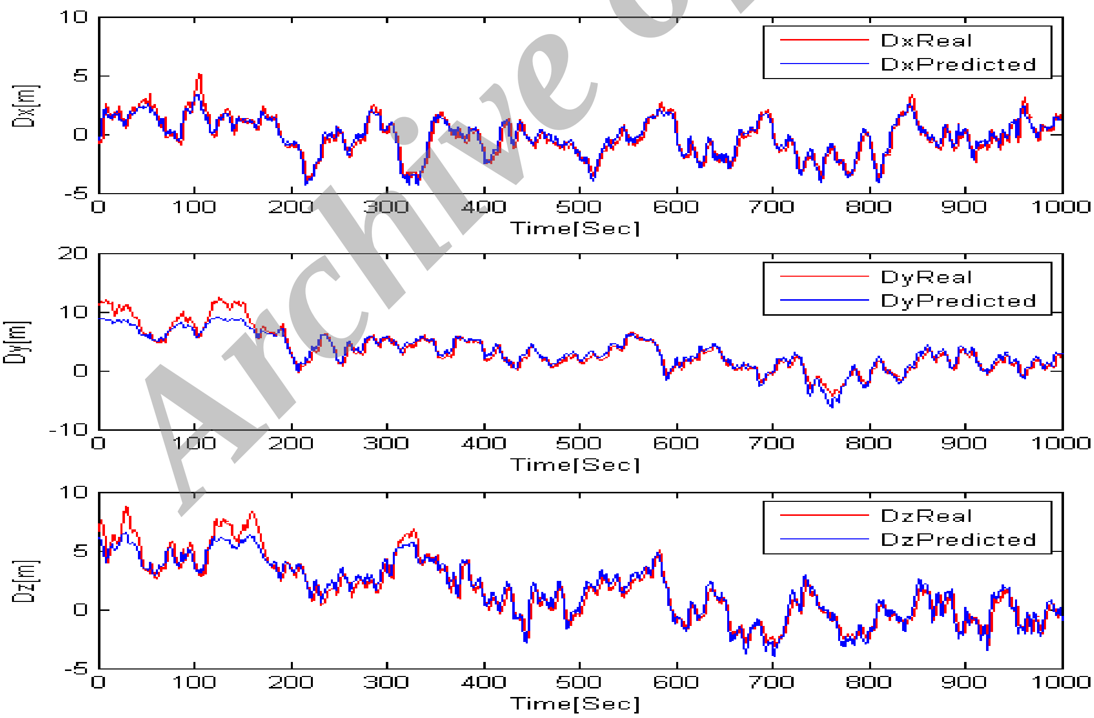


***Figure 4. Using MLP NN trained by ICA learning algorithm for prediction of DGPS corrections (SA off)***

***Table 2. Performance evaluations of proposed NNs trained by BP and ICA for DGPS corrections prediction (SA on)***

| Learning Algorithms | Parameters | X Component | Y Component | Z Component |
|---|---|---|---|---|
| BP | Max | 5.8722 | 4.1216 | 6.2002 |
| | Min | -5.8251 | -5.4896 | -7.6102 |
| | Average | 0.0188 | 0.0303 | -0.0502 |
| | RMS | 1.1288 | 1.2415 | 1.0747 |
| ICA | Max | 1.7240 | 1.3349 | 1.3253 |
| | Min | -2.6222 | -1.8124 | -1.5444 |
| | Average | -0.1556 | 0.0235 | 0.0047 |
| | RMS | 0.6543 | 0.3817 | 0.3327 |

***Table 3. Performance evaluations of proposed NNs trained by BP and ICA for DGPS corrections prediction (SA off)***

| Learning Algorithms | Parameters | X Component | Y Component | Z Component |
|---|---|---|---|---|
| BP | Max | 1.5017 | 1.7745 | 1.5576 |
| | Min | -1.6887 | -1.6569 | -1.5302 |
| | Average | -0.0031 | 0.0160 | 0.0099 |
| | RMS | 0.4249 | 0.4697 | 0.4247 |
| ICA | Max | 1.6203 | 1.6857 | 1.3619 |
| | Min | -2.9247 | -3.6669 | -2.5310 |
| | Average | 0.0032 | 0.0040 | 0.0037 |
| | RMS | 0.5364 | 0.3380 | 0.3291 |

The comparison of total RMS error in DGPS corrections prediction using NNs trained by the two learning algorithms based on the BP and ICA showed in Table 4.

***Table 4. Performance comparison of proposed NNs trained by BP and ICA for DGPS corrections prediction***

| Learning Algorithms | Accuracy with SA on | Accuracy with SA off |
|---|---|---|
| BP | 1.9926 | 0.7626 |
| ICA | 0.8273 | 0.7143 |

Table 4 demonstrates that total RMS error of prediction by ICA trained NN is lower than BP trained. Therefore, the NN trained by learning algorithm based on the ICA has better accuracy for DGPS corrections prediction; so that the total RMS error reduces to less than 0.8273 meter with SA on and 0.7143 with SA off.

Table 5 shows the comparing DGPS corrections prediction accuracy using NNs trained by the four learning algorithms based on the BP, Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and ICA.

***Table 5. Comparing DGPS corrections prediction accuracy using proposed NN trained by BP, GA, PSO and ICA***

| Prediction Method | Accuracy |
|---|---|
| NN trained by BP | 1.96 |
| NN trained by GA | 1.34 |
| NN trained by PSO | 1.32 |
| NN trained by ICA | 0.83 |

As shown in Table 5, the NN trained by learning algorithm based on ICA has better accuracy for DGPS corrections prediction, since RMS error of prediction in this method is lower than other evolutionary methods.

There are a few papers (for examples [4] and [10]) that estimate the pseudo-range DGPS corrections using NNs. The proposed NNs in this paper have more accuracy than them.

## 4. Conclusions

There are a lot of errors associated with the navigation solution of the GPS that has been one of the main systems for navigation aids. In order to reduce these and other errors (satellite clock error, ephemeris error, multi-pathing, tropospheric delays, ionospheric delays, SA error and etc.), DGPS correction signal is required which allows civilian users of GPS to achieve greater accuracy than just the GPS signal. In this paper, an innovative methodology based on NN has been developed which predicts pseudo-range corrections and compensates the GPS data in near time. The accuracy of the pseudo-range predictions is estimated and maintained by the algorithm. The technique's experimental results demonstrate that the prediction method based on NNs allows the rover to effectively compensate for the communication latency. These experiments also confirm that the prediction total RMS errors using NN based on ICA learning algorithm are 0.8273 meter and 0.7143 meter, before and after SA, respectively.

## 5. References

[1] P. J. Mumford, "Relative Timing Characteristics of the One Pulse Per Second (1PPS) Output Pulse of Three GPS Receivers," The 6th International Symposium on Satellite Navigation Technology Including Mobile Positioning & Location Services, pp.1-10, 2003.

[2] K. Kawamura and T. Tanaka, "Study on the Improvement of Measurement Accuracy in GPS," SICE-ICASE International Joint Conference, pp.1372-1375, 2006.

[3] M. R. Mosavi, "Comparing DGPS Corrections Prediction using Neural Network, Fuzzy Neural Network, and Kalman Filter," GPS Solutions, Vol.10, No.2, pp.97-107, 2006.

[4] A. Indriyatmoko, T. Kang, Y. J. Lee, G. I. Jee, Y. B. Cho and J. Kim, "Artificial Neural Networks for Predicting DGPS Carrier Phase and Pseudo-Range Correction," GPS Solutions, Vol.12, No.4, pp.237-247, 2008.

[5] M. R. Mosavi, "Wavelet Neural Network for Corrections Prediction in Single-Frequency GPS Users," Neural Processing Letters, Vol.33, No.2, pp.137-150, 2011.

[6] E. A. Gargari and C. Lucas "Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition," IEEE Congress on Evolutionary Computation, pp.4661-4667, 2007.

[7] M. R. Mosavi, H. Nabavi and A. Nakhaei, "Neural Technologies for Precise Timing in Electric Power Systems with a Single-Frequency GPS Receiver," Wireless Personal Communications, Vol.75, No.2, pp.925-941, 2014.

[8] M. Kh. S. Alsmadi, Kh. B. Omar and Sh. A. Noah, "Back Propagation Algorithm: The Best Algorithm Among the Multi-Layer Perceptron Algorithm," International Journal of Computer Science and Network Security, Vol.9, No.4, pp.378-383, 2009.

[9] Z. Zainuddin, N. Mahat and Y. A. Hassan, “Improving the Convergence of the Backpropagation Algorithm using Local Adaptive Techniques,” International Journal of Computer, Information, Systems and Control Engineering, Vol.1 No.1, pp.1-4, 2007.

[10] J. Sang, K. Kubik and L. Zhang, “Prediction of DGPS Corrections with Neural Networks,” IEEE Conference on Knowledge-based Intelligent Electronic Systems, pp.355-361, 1997.