Case-Study Plenary Paper

NEURAL NETWORK BASED ORBIT PREDICTION FOR A GEOSTATIONARY SATELLITE

A. T. Kutay*, Y. Tulunay†, E. Tulunay‡, O. Tekinalp§

Abstract

An artificial Neural Network (NN) model was developed to estimate the semi-major axis (a), the eccentricity (e) and the inclination (i) of a geostationary satellite orbit. To facilitate a comparison between the NN model developed herewith and a real case, the TÜRKSAT 1B geostationary satellite has been taken as example. A code that numerically solves the parameters of the TÜRKSAT's orbit, namely METUAEE1, is used to generate the training data for the NN model and to evaluate its performance. A Multi-Layer Perceptron (MLP) *type NN model is constructed and it is trained by the Steepest Descent algorithm with one dimensional search. There are two different approaches studied for the estimation of the orbit parameters. In the first one the NN is trained 'on-line' while in the second one 'off-line' training is performed. Copyright ©2001 IFAC*

1. Introduction

In the classical sense, the orbit determination process consists of estimating the parameters that define an orbit. In other words, it means to obtain some solutions of a mathematical orbit equation at specified time intervals by means of an available numerical or analytical method. The principal of the whole exercise is to make a decision of some realistic and appropriate models by the help of realistic and appropriate assumptions when the observational data are not available throughout the orbit. Once the assumptions are established, the physical principles which are the two-body motion for an earth bound satellite and the effective perturbations are applied in order to model the orbital motion as a set of differential equations relating the accelerations to the applied forces and torques [2].

The METUAEE1 computer code written in FORTRAN 77 was developed by Şakacı to simulate the orbital motion of a geostationary earth satellite, namely the TÜRKSAT 1B satellite by predicting its future path [1]. In METUAEE1, a special perturbation technique, namely the Cowell's method was employed where the differential equations of motion with all the effective perturbative accelerations taken into account are directly integrated [3]. The results of the METUAEE1 were validated by the results of SORBET, the commercial orbital software used by the TÜRKSAT Ground Control Station developed by Aerospatiale. The results of the METUAEE1 exhibit a high correlation with the results of the SORBET. For the orbit parameters investigated herewith, namely the semi-major axis (a), the eccentricity (e) and the inclination (i), the correlation coefficients between SORBET and METUAEE1 results are all greater than 0.99 for a simulation covering a time interval of two weeks [1].

The classical simulation tools need the accurate modeling of the perturbations for accurate results. The assumptions made in the mathematical modelling of the perturbative forces and ignorance of some of these forces limit the accuracy of the simulation. Therefore some fashionable *Intelligent Control* methods may be adopted.

A literature survey covering the last ten years or so showed that there is very little work on the relevant topics and in fact to our knowledge there has been no direct NN model to predict the geosynchronous orbit parameters reported. A Multi Layer Perceptron (MLP) type NN developed to predict the three parameters of the orbit a, i and e will be presented herewith.

In the artificial intelligence approach presented in this paper an NN based model of the orbit is aimed to be obtained. With this model it is possible to predict any one of the parameters of an orbit at a specified

* B. S., Middle East Technical University, Aeronautical Engineering Department, 06531 Ankara, Turkey
Phone: ++(90) (312) 210 4278, Fax: ++(90) (312) 210 1272, E-mail: tkutay@ae.metu.edu.tr
† Professor, Dean of the Istanbul Technical University, Faculty of Aeronautics and Astronautics, 80626 Istanbul, Turkey
Phone: ++(90) (212) 285 3341, Fax: ++(90) (212) 285 3139, E-mail: ytulunay@hidiv.cc.itu.edu.tr
‡ Professor, Middle East Technical University, Electrical and Electronics Engineering Department, 06531 Ankara, Turkey
Phone: ++(90) (312) 210 2335, Fax: ++(90) (312) 210 1261, E-mail: ersintul@rorqual.cc.metu.edu.tr
§ Associate Professor, Middle East Technical University, Aeronautical Engineering Department, 06531 Ankara, Turkey
Phone: ++(90) (312) 210 4278, Fax: ++(90) (312) 210 1272, E-mail: tekinalp@rorqual.cc.metu.edu.tr

time within the vicinity of the time interval chosen for training by just giving that specific time. In other words the NN model provides the orbit parameters corresponding to a time input. In the classical simulations however, in order to obtain the parameters at a given time instant, the initial conditions at a prior time must be available. Then the evolution of the parameters in time will be obtained by solving the differential equations of motion numerically with the given initial conditions. To have the parameter at a specific time instant, the solution of the equations of motion must be extended to that instant. The advantage of the NN model is that the orbit of the spacecraft at a time in the future will be estimated without considering the differential equations of motion and the perturbative effects like the moon and the sun.

The NN model obtained here is mainly based on some synthetic data generated by the METUAEE1. Whereas, the NN model would have been more realistic if real orbit data were employed. However our approach is general and can be used for any relevant data.

2. The Classical Orbit

The motion of a spacecraft is specified by its position, velocity, attitude and attitude motion. The first two quantities describe the translational motion of the center of mass of the spacecraft and are the subjects of celestial mechanics. The foundation of celestial mechanics is simply the set of Newton's Law of Motion plus his Universal Law of Gravitational Attraction and with the addition of a set of empirical laws of Kepler which describe the planetary motion [4].

Coordinate systems differ in the orientation of the fundamental plane and of the principal axis, and in the origin for coordinates. Each coordinate frame has a particular property which makes it appropriate to a limited number of applications. In the satellite orbital analysis, 'Latitude-Longitude Coordinate System' is referred. In this coordinate system, on the earth, the location of an orbit is given by its two angular coordinates and its altitude above/below the adopted reference ellipsoid. The origin of the system is the earth center. Therefore, it is a geocentric system as illustrated in Figure 2-1. The fundamental plane is the equator, and the principal axis direction is the same as the direction of the Greenwich meridian from the geocenter [4].

In Figure 2-1, ϕ' is the geocentric latitude measured perpendicular to the equatorial plane, between the line joining the observer to the geocenter and the equatorial plane. The other angular coordinate to define the location of an observer on the earth is the east longitude λ_E. It is measured towards the east in

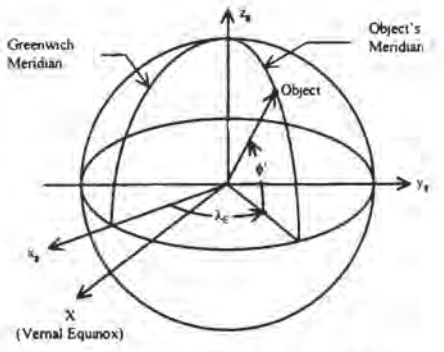

Figure 2-1 Latitude-longitude coordinate system

the equatorial plane, between the observer's meridian and the Greenwich meridian. And, f is the altitude which is the distance of the observer above/below the reference ellipsoid [4].

Five independent quantities called 'orbital elements' are sufficient to completely describe the size, the shape and the orientation of an orbit. And a sixth element is required to set the position of a spacecraft along the orbit at a particular time. These six elements are defined by the help of Figure 2-2 and Figure 2-3 as follows [1]:

1. **a, semi-major axis:** a constant defining the size of the orbit.
2. **e, eccentricity:** a constant defining the shape of the orbit.
3. **i, inclination:** the angle between the orbital plane of the spacecraft and the equatorial plane.
4. **Ω, right ascension of the ascending node:** the angle between the vernal equinox direction and the line defining the intersection of the equatorial and the orbital planes as a point in the orbital plane (ascending node) passes through the equator from south to the north.
5. **ω, argument of perigee:** the angle, in the plane of the satellite's orbit between the ascending node and the perigee point (the nearest point of the orbit to the attractive center, namely the

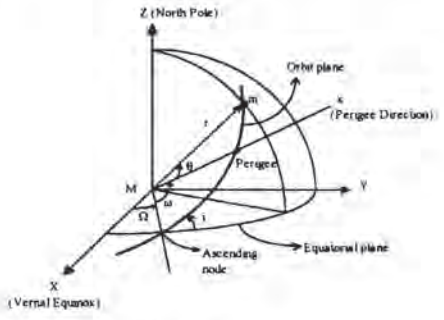

Figure 2-2 Orbital elements in space

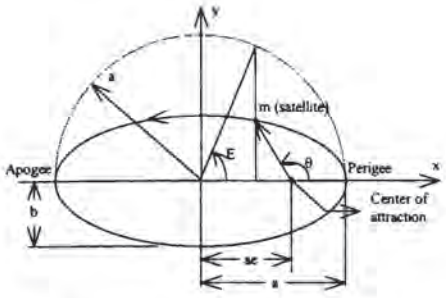

Figure 2-3 Orbital elements in orbital plane

Apogee : Most distant point to earth center
Perigee : Closest point to earth center
a : Semi-major axis, the distance from apogee to conic center
b : Semi-minor axis

earth) direction.

6. **θ, true anomaly:** the angle, in the plane of the satellite's orbit between the perigee direction and the line connecting the satellite to the focus of the orbit. For two body motion, it is the only time dependent variable among the others.

Three different angles are defined to set the position of an spacecraft on its orbit:

True anomaly (θ): Angle between the satellite direction and the perigee direction.
Mean anomaly (M): Angle measured from perigee direction. Proportional to time. M = nt (n = Mean motion = 360 ° / T)
Eccentric anomaly (E): Intermediate variable relating the other two angles, measured at the orbit center, between perigee and the projection of the satellite on a circle of radius 'a' [1].

Ideal geostationary (GEO) orbit is a circular, equatorial and synchronous orbit. The GEO orbit has a zero eccentricity since it is a circular orbit. The revolutional period of a satellite on the GEO orbit must be equal to that of earth around its axis, so there is a zero relative velocity between earth and the satellite. GEO orbit altitude must be chosen as 35,786 km (from ground) to obtain synchronism with respect to earth as it follows from the Kepler's 3rd law [5].

3. The NN Model

A modular and general purpose computer program is developed to simulate a Multi Layer Perceptron (MLP) type Neural Network (NN) (Figure 3-1) [6]. The parameters that define the structure of the network; the number of inputs (maximum 20), the number of layers (maximum 5), the number of

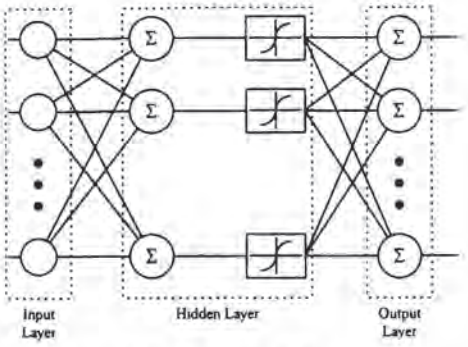

Figure 3-1 MLP type NN structure

neurons in each layer (maximum 20) can be adjusted. The program lets user change these parameters at the beginning of the execution. Number of iterations to be performed can also be adjusted at the beginning. Tangent hyperbolic type sigmoid function is used for the nonlinear elements in the network (Figure 3-2). For the training of the network, Steepest Descent algorithm is employed. Program also utilizes one-dimensional search to choose the step length [7].

3.1 Preparation of the Training Data Set for the NN

From the five parameters defining the size, the shape and the orientation of the orbit, following three contribute to the longitude and the latitude of a spacecraft: the semi-major axis (*a*), the eccentricity (*e*) and the inclination (*i*). Therefore these parameters are studied in this work.

The semi-major axis is ideally 42,161,000 m from the center of the earth. The order of magnitude of the semi-major axis is very large to be processed in the NN due to the nonlinear function used in the neurons (Figure 3-2). The variations in the input parameters on the order of 10^7 can not be sensed by the neurons directly. Therefore, a transformation is performed on the semi-major axis values. The semi-major axis of a geostationary telecommunication satellite may deviate ±10,000 m from the nominal value of 42,161,000 m. Instead of the variation of the semi-major axis, its deviation from its nominal value is used. This resulted in a parameter varying on the order of 10^4. Then it is scaled by a factor of 10^{-3} and a data set sensible for the NN is obtained. Similarly, the data of the inclination (*i*) and the eccentricity (*e*)

$$\tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

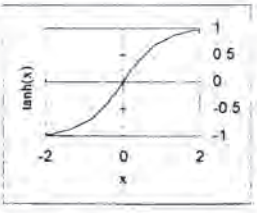

Figure 3-2 Nonlinear function used in the neurons

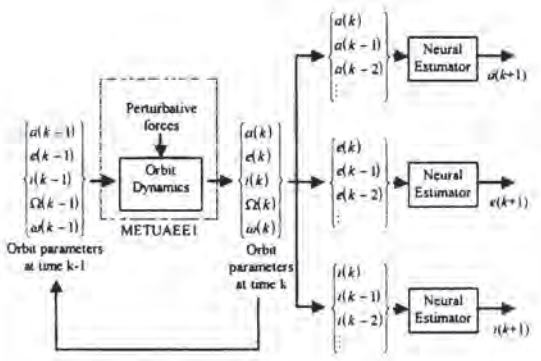

Figure 3-3 Flowchart of on-line trained NN estimator

are scaled by factors of 10^3 and 10 respectively without shifting.

3.2 On-line Training Approach

The NN used in this approach is actually a neural estimator. A small (in terms of number of neurons used) NN is used on-line to predict the output at the next sampling time. The orbit is modelled as a discrete time system. At each sampling interval, the estimator is trained with the current and past data and then it gives an estimation for the output at the next sampling time (Figure 3-3).

To perform the simulation the orbit parameters at time $k = 0$ are required (initial conditions). Then the evolution of the parameters in time is calculated by the METUAEE1 by solving the equations of motion numerically. The results of the METUAEE1 code is then used to prepare the training data set for the NN based simulation program. This program will be referred as ATKNN from now on. In ATKNN a separate NN is used to estimate each parameter. Every time the NNs are trained with a number of past values of the parameters. After a certain number of iterations they give an estimate for the value of the parameter at the next sampling time. Since in this method the NNs are trained on-line at each time step, the method is referred as *on-line training method*. Depending on the time step chosen for the simulation, i.e., how frequently the samples are taken, the accuracy of the estimation changes as one can expect intuitively.

The training data set for each of the NNs is as follows:

$$\left\{ \left[\underbrace{\begin{pmatrix} p(k-1) \\ p(k-2) \\ p(k-3) \\ \vdots \\ p(k-n) \end{pmatrix}}_{\text{Input vector}}, \underbrace{p(k)}_{\text{Desired output}} \right], \left[\begin{pmatrix} p(k-2) \\ p(k-3) \\ p(k-4) \\ \vdots \\ p(k-n-1) \end{pmatrix}, p(k-1) \right], \ldots, \left[\begin{pmatrix} p(k-N) \\ p(k-1-N) \\ p(k-2-N) \\ \vdots \\ p(k-n-N+1) \end{pmatrix}, p(k-N) \right] \right\} \quad (1)$$

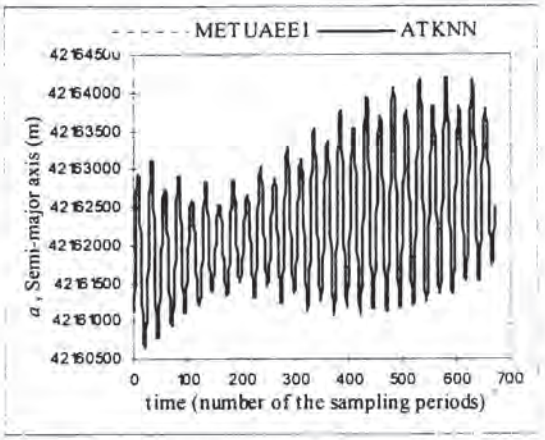

Figure 3-4 Variation of the semi-major axis in 14 days calculated by the METUAEE1 and estimated by the ATKNN with arbitrarily set parameters

where $p(k)$ represents any one of the parameters a, e and i, n is the number of the elements in the input vector and N is the number of input-output pairs in the training data set. This data set teaches the NN the relation between the past values of the parameter and its value at the next simulation time. After the training with this set is performed for a few iterations, following input vector is given to the NN and then it is expected to give an estimation for the parameter at time $(k+1)$:

$$\text{Input vector: } \begin{pmatrix} p(k) \\ p(k-1) \\ p(k-2) \\ \vdots \\ p(k-n+1) \end{pmatrix} \quad \text{output: } p(k+1) \quad (2)$$

The procedure described above provides an estimate of the parameters a, e and i at one time step ahead of the current time. A simulation is performed with $n = 4$ and $N = 5$ for a period of 14 days. All the orbit parameters are set to their nominal values for the TÜRKSAT 1B satellite initially. 30 iterations are performed for the training of the NNs at each simulation interval. Figure 3-4 shows the variation of the semi-major axis of the TÜRKSAT 1B satellite calculated by the METUAEE1 and estimated by the ATKNN. On this graph at each time instant k, the value of a calculated by the METUAEE1 at k and estimated by the ATKNN at k - 1 is plotted. At any time k during the simulation, the ATKNN is trained with the previous data and it estimates the value of the parameter at time k + 1. The value of the parameter at k + 1 calculated by the METUAEE1 is already available in the training data set. Then these two values are marked on the graph. Figure 3-4 is obtained by repeating the above procedure along the whole simulation interval. The horizontal axis in the graph is time in terms of sampling periods. For this

simulation sampling period is chosen as 30 minutes. The root mean square error and the normalized root mean square error are chosen for the performance measure of the ATKNN.

$$rms\ error = \sqrt{\frac{\sum_{k=1}^{\text{Number of data}} \left(p(k) - p_{estimated}(k)\right)^2}{\text{Number of data}}} \qquad (3)$$

$$normalized\ rms\ error = \sqrt{\frac{\sum_{k=1}^{\text{Number of data}} \left(\frac{p(k) - p_{estimated}(k)}{p(k)}\right)^2}{\text{Number of data}}}$$

For this simulation the rms error and the normalized rms error turned out to be 29.91 meters and $7.1*10^{-7}$ respectively. This number shows how accurately the value of the semi-major axis (actually its deviation from the nominal value) is estimated half an hour in advance.

The error between the values of a estimated by the ATKNN and calculated by the METUAEE1 is so small that they superimpose each other. In order to illustrate how well the ATKNN estimates the semi-major axis Figure 3-5 is presented. Figure 3-5 shows the cross correlation curve between the ATKNN and the METUAEE1. The correlation coefficient between the ATKNN's estimation of the semi-major axis and the METUAEE1's results is calculated as 0.9996837.

Since one-dimensional search is employed in the training equations, increasing the number of iterations guarantees the decrease of the error in learning the training data set [7]. However, although the learning of the training data set improves as the number of iterations is increased, generalization capability of the NN begins to decrease after a certain limit, which means that the estimation error

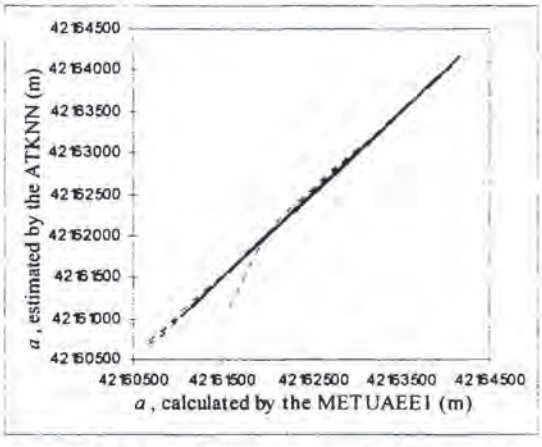

Figure 3-5 Cross correlation curve between the METUAEE1 and the ATKNN with arbitrarily set parameters

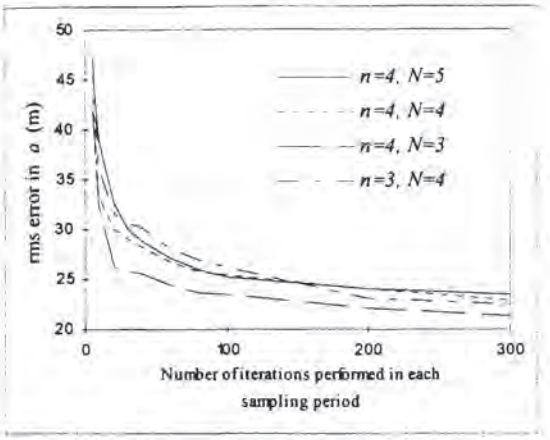

Figure 3-6 Effect of the number of iterations on the average estimation error

begins to increase [6].

In order to determine the optimum parameters of the NN and the training data set many simulations are performed each time with a different set of parameters (n, N, number of iterations, number of layers and number of neurons). The parameters are changed systematically to observe their effects on the performance. Figure 3-6 and Figure 3-7 are obtained and used to choose the number of iterations to be performed and the parameters of the training data set respectively. During this exercise, the semi-major axis is considered only. The eccentricity (e) and the inclination (i) are the two other parameters defining the same physical phenomenon as a and are under the effects of the same forces and moments. So, it is assumed that they have the similar characteristics as the semi-major axis (a) and thus the optimum parameters of the NN and the training data set by considering the semi-major axis are also the optimum parameters for the estimation of e and i.

Figure 3-6 shows how the error changes with the number of iterations. Simulations with different number of iterations with different training data sets are performed. It is seen that the error decreases steeply until 50 iterations and this is almost independent of the parameters of the training data set . The decrease in the rms error after 50 iterations is not very significant. That is, it is not worth to make that many iterations since increasing the number of iterations would increase the simulation run time. By making use of this piece of information it is decided to make 60 iterations at each simulation time for all of the three NNs.

After choosing the number of iterations, next step was to decide on the training data set parameters, n, the number of elements in the input vector and N, the number of input/output pairs in the training data set as appeared in Eqn. (1). Keeping the number of

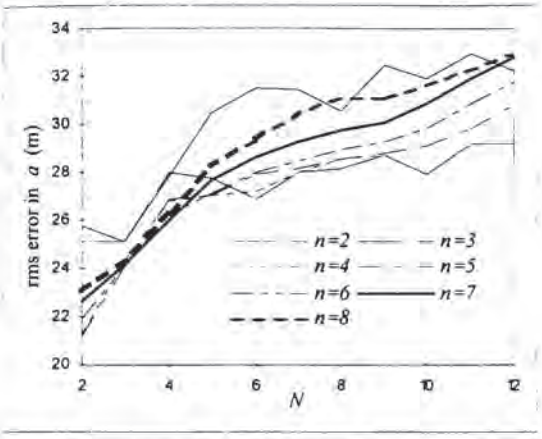

Figure 3-7 Effect of the training data set parameters on the average estimation error

iterations constant at 60, Figure 3-7 is obtained by varying n from 2 to 8 and N from 2 to 12. As it is seen from this figure the minimum error is achieved with $N = 2$ and $n = 4$.

Keeping N, n and the number of iterations to be performed constant at the values decided above, the number of neurons in the hidden layer is varied. Figure 3-8 shows the variation of the rms error with the number of neurons. It is seen from this figure that for $N = 2$ and $n = 4$, 10 neurons give the minimum error. In the previous simulations the number of neurons was chosen arbitrarily as 10 and this number turned out to be the optimum number. From this figure once again it is seen that the values selected for n and N in the previous study are the optimum values.

The last parameter to be selected to set the structure of the NN is the number of the hidden layers. All of the previous simulations were performed by using an NN with one hidden layer containing 10 neurons. To see the effect of the number of layers on the

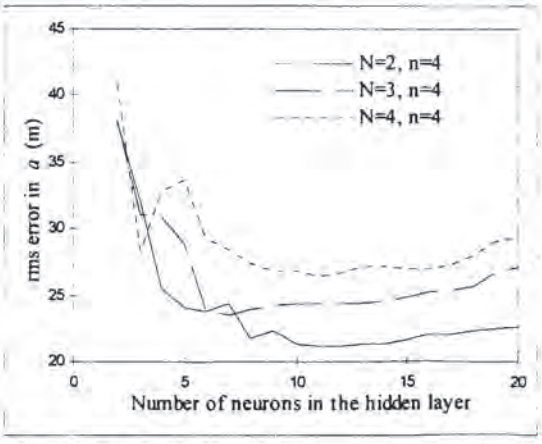

Figure 3-8 Variation of the rms error with the number of neurons

generalization capability of the NN, a two layer NN is used and its results are compared with those of the one layer NN. For these simulations the parameters that give the minimum rms error with the one layer NN are used. Figure 3-9 shows the estimation performance of a two layer NN as compared to a single layer one. In this figure, the horizontal axis is the number of neurons used and the vertical axis is the rms error. For the NN with two hidden layers, same number of neurons are used at both layers. As the number of layers is doubled, the number of neurons in the NN is also doubled if the same number of neurons are used in the two layers. Doubling the number of neurons means that the number of parameters to be adjusted (weights and biases) in the NN during training is also doubled. One intuitively expects that increasing the number of neurons in an NN will increase its learning and generalization capability. However the result is not that. In the simulations with one layer NN 60 iterations were performed. When the number of parameters to be adjusted in training is increased, more iterations may be required to be performed. Considering this may be the reason for the degradation of the performance the number of iterations are increased up to 500. However, as it is seen in the Figure 3-9 the estimation performance of the double layer NN is worse than that of the single layer NN although much more iterations are performed. So, single layer NNs are decided to be used in the ATKNN as they give better performance and need less time for training. After the above studies the parameters presented in Table 3-1 are adopted to be used in the rest of this work.

N	2
n	4
Number of hidden layers	1
Number of neurons in the hidden layer	10
Number of iterations	60

Table 3-1 The NN parameters

After constructing the NN model ATKNN, a simulation is performed with the parameters presented in Table 3-1 and the estimations for a, e and i are obtained. The curves obtained for a, e and i versus time for the 14 days period are presented in Figure 3-10, Figure 3-11 and Figure 3-12 respectively in comparison with the METUAEE1 results. The rms error for the semi-major axis turned out to be 21.28 m which was 29.91 m in the first run.

The normalized rms errors in the estimation of the orbit parameters are given in Table 3-2 together with the rms errors and the correlation coefficients. In this table it is seen that the normalized error for a is much

Orbit parameter	rms error	Normalized rms error	Correlation coefficient
a	21.28 m	$5.05*10^{-7}$	0.9996837
e	$3.24*10^{-6}$	$2.17*10^{-2}$	0.9970946
i	$2.09*10^{-3}$ deg	$3.54*10^{-2}$	0.9892513

Table 3-2 The rms errors, the normalized rms errors and the correlation coefficients in the estimation of a, e and i

less than those for e and i. This is because the deviation from the semi-major axis is investigated rather then the semi-major axis itself. Then the error in the deviations is normalized with respect to the semi-major axis. That is, a quantity on the order of 10^3 (the deviations from the nominal value) is normalized with respect to a quantity that is on the order of 10^7 (the semi-major axis). This causes the normalized rms error values for the semi-major axis to appear less than those for the eccentricity and the inclination approximately with a factor of 10^{-4}. Actually, the performances of the ATKNN on the estimation of a, e and i are close to each other.

In these simulations, sampling interval was chosen as 30 minutes. So time $k + 1$ corresponds to the time 30 minutes ahead of time k and thus by choosing the sampling period as 30 minutes the values of the orbital parameters could be estimated 30 minutes in advance. In practice however, the values of the parameters at time instants later than half an hour may be needed. The easiest way to obtain this may be to increase the sampling interval. However the sampling interval must be small compared to the period of the parameter and cannot be increased freely. The period of the variations in the semi-major axis is around 12 hours and in the eccentricity is around 24 hours. The variation of the inclination is not periodic in the 14 days interval. So, the sampling points must be dense enough in the 12 hours period

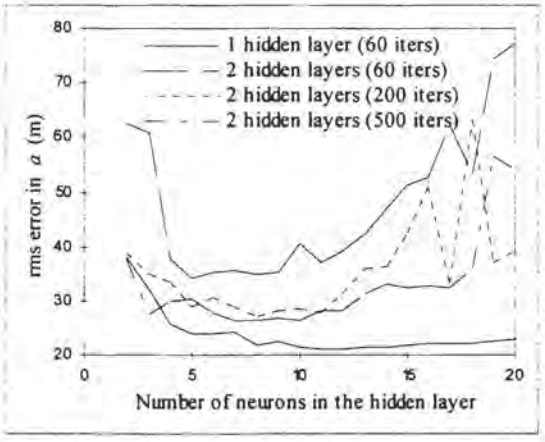

Figure 3-9 Variation of the rms error with the number of neurons and the number of layers

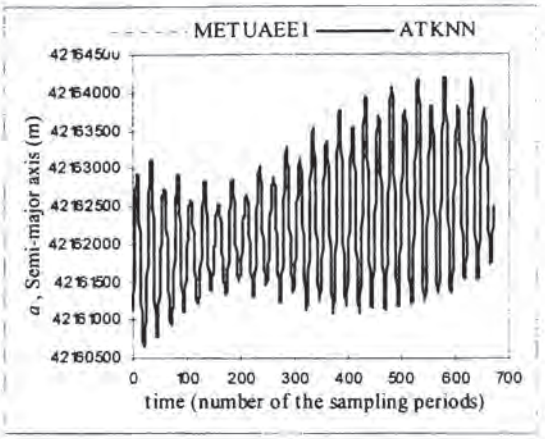

Figure 3-10 Variation of the semi-major axis in 14 days calculated by METUAEE1 and estimated by ATKNN with the on-line training method

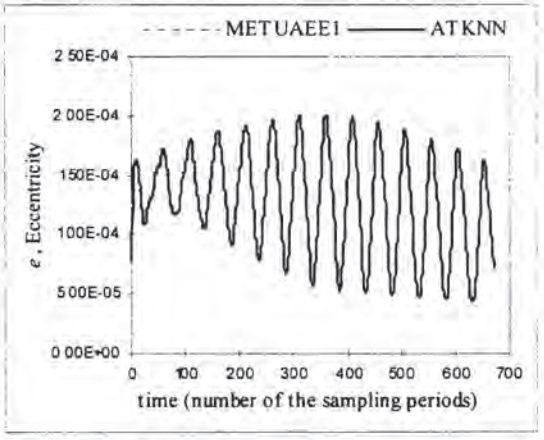

Figure 3-11 Variation of the eccentricity in 14 days calculated by METUAEE1 and estimated by ATKNN with the on-line training method

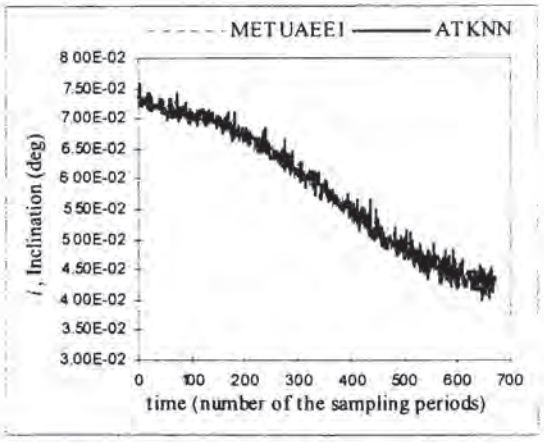

Figure 3-12 Variation of the inclination in 14 days calculated by METUAEE1 and estimated by ATKNN with the on-line training method

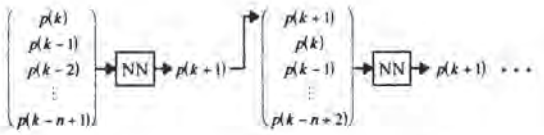

Figure 3-13 Propagation of the estimations

so that the variation of all of the three parameters will be sensed by the ATKNN. There are 24 sampling points in one period of the semi-major axis with the sampling interval equal to 30 minutes. If the sampling interval is increased, that is, less number of samples are taken, the characteristic of the semi-major axis will begin to disappear and this will naturally decrease the accuracy of the estimation. Table 3-3 exhibits the rms errors in the estimated semi-major axis values one sampling period in advance with different sampling periods.

A second way to obtain future estimations is to make successive estimations for the time instants $(k + 1)$, $(k + 2)$, $(k + 3)$, ... at each sampling interval. First train the NN with the data set in (1) and obtain an estimation for $p(k + 1)$ by using the input vector in (2). Then construct a new input vector by adding the estimated value of $p(k + 1)$ and by using this input vector obtain an estimation for $p(k + 2)$. Repeating this procedure to estimate the parameters at time $(k + 3)$, $(k + 4)$, ... (
Figure 3-13).

Keeping all the parameters at the values shown in Table 3-1 and the sampling period at 30 minutes, simulation program is run again and the estimations are propagated up to $(k + 12)^{th}$ instant. Figure 3-14 shows the normalized rms errors in the estimation of the eccentricity and the inclination as the estimation range is enlarged. In this figure, the horizontal axis shows how far is the estimation time from the current time in terms of the sampling periods. As expected, the rms error increases as the estimation window is enlarged.

In the first step of the estimation where the vector

Sampling period (minutes)	rms error (meters)	Normalized rms error
15	14.13	$3.35*10^{-7}$
30	21.28	$5.05*10^{-7}$
45	39.17	$9.29*10^{-7}$
60	62.33	$1.48*10^{-6}$
75	95.11	$2.26*10^{-6}$
90	136.86	$3.25*10^{-6}$

Table 3-3 The rms and the normalized rms errors in the estimation of a at one sampling time ahead of the current time for different sampling periods.

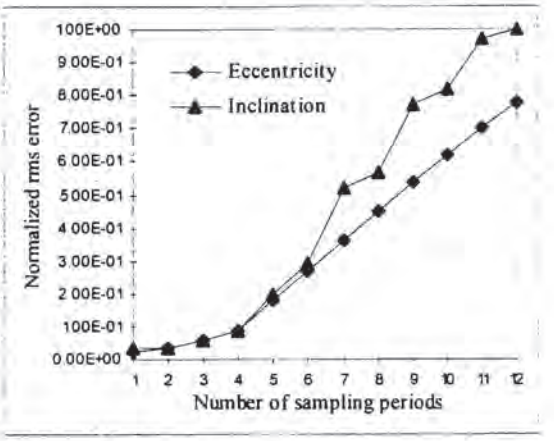

Figure 3-14 The variation of the rms error as the estimations are propagated into the future

$[p(k), p(k - 1),..., p(k - n + 1)]$ is given to the NN (Figure 3-13), $p(k + 1)$ is estimated with some error due to the training and generalization capabilities of the NN. In the second step where $p(k + 2)$ is to be estimated, $p(k + 1)$ estimated in the previous step is added to the input vector. So, in the estimation of $p(k + 2)$, the generalization error is added to the error in the input vector and thus the error in the estimation of $p(k + 2)$ becomes greater than the error in the estimation of $p(k + 1)$. In the next step $p(k + 2)$ and $p(k + 1)$, two estimated and thus two inaccurate elements appear in the input vector. The inaccuracy of the input vector and the generalization error makes the error in the third step greater than the previous ones and consequently the error increases at each step (Figure 3-14).

As it is seen from Figure 3-14, the normalized rms error in the estimations for the $(k + 12)^{th}$ instant reaches 100%. After four sampling periods (two

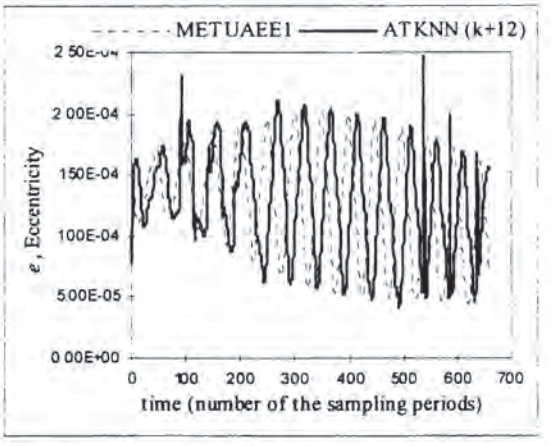

Figure 3-15 ATKNN's estimation of the eccentricity twelve sampling periods (six hours) in advance compared with the METUAEE1's calculations

hours) the error in the estimations exceed 10%. Figure 3-15 shows ATKNN's estimation of the eccentricity twelve sampling periods (six hours) in advance compared with the METUAEE1's calculations.

With the propagation of the estimations technique the parameters at later than one sampling time ahead of the current time can be estimated. Previously the sampling period was increased in order to estimate the parameters at times farther from the current time. Table 3-4 shows the comparison of the two techniques, the increase of the sampling period and the propagation of the estimations. In this table the first column shows the time the estimations are made in advance. This time is reached by two different ways in the two techniques. In the first one, the increase of the sampling period, this time is reached in one sampling period by increasing the sampling period. For example, to estimate the semi-major axis one hour in advance the sampling period is chosen as 60 minutes. In the second technique the sampling period is kept constant at 30 minutes. In this method to estimate the semi-major axis one hour in advance, the estimations are propagated one more sampling time and the value at time $(k + 2)$ is taken. It is concluded from this table that increasing the sampling period in order to estimate the semi-major axis earlier gives better results than propagating the estimations. This naturally has a limit, that is, the sampling period can never be selected greater than the period of the parameter investigated. For example while the semi-major axis changes with a period of 12 hours the sampling period can not be selected close to or greater than 12 hours.

Time (min)	rms error in a (meters)	
	Increase of the sampling period	Propagation of the estimations (T=30)
30	21.28	21.28
60	62.33	71.21
90	136.86	157.18

Table 3-4 Comparison of the increase of the sampling period and the propagation of the estimations technique

3.3 Off-line Training Approach

In this method, the NNs are trained with the prepared data set and after the training is complete they estimate a value for any time within the vicinity of the trained time interval by using their generalization capability. A data set including the values of the studied parameter for a certain time interval (typically a few weeks) is taught to the NN. After the NN is trained, it can estimate the value of the parameter at a time instant close enough to the trained time interval. As in the on-line training approach accuracy of the estimations depend on the closeness of the estimation time to the trained time interval. In the first method only a few samples of the parameter (typically around 10) is taught to the NN at each sampling period and so it can estimate the value of the parameter with an acceptable error (for example less than 10%) only within the next few sampling periods. To have a curve of the parameter for a period of two weeks for example, one has to start the simulation from the beginning of the time interval and propagate through the interval step by step at each step training the NN with the previous data and estimating the next few points.

In the second method that will be explained in this section, the NNs are trained once with a much more rich data set not only including a few data points but covering a much wider time interval including hundreds of data points. After they are trained, that is, the NN weights and biases are set, they can estimate the neighborhood of the trained interval. Depending on the richness of the training data set, the time limit that they can estimate with an acceptable error may extend to days.

As it was done in the on-line training method, in the off-line training method also a simulation is made with arbitrary NN parameters without making any optimization study on the choice of these parameters.

In the first run, a training data set of the semi-major axis of a two weeks period is prepared. The sampling period is chosen as 30 minutes again. The information included in the training data set is the most important difference between the on-line and the off-line training methods. In the on-line training method, only the past values of the studied parameter (a, e or i) were used in the input vector of the NNs as showed in Eqn. (1) and using this information the values at the next few sampling points were estimated. In other words, the past-future relation of the parameters were taught to the NNs. In the off-line training method however, the time dependence of the parameters are taught. That is, while training the NNs instead of the past values of the parameter, the time they belong to is given in the input. Four different parameters are given to specify the time, which are the Mean Julian Date, month, day and minute. The training data set is given as:

$$\left\{ \left[\underbrace{\begin{pmatrix} MJD(1) \\ month(1) \\ day(1) \\ minute(1) \end{pmatrix}}_{\text{Input vector}}, \underbrace{p(1)}_{\text{Desired output}} \right], \left[\begin{pmatrix} MJD(2) \\ month(2) \\ day(2) \\ minute(2) \end{pmatrix}, p(2) \right], \ldots, \left[\begin{pmatrix} MJD(N) \\ month(N) \\ day(N) \\ minute(N) \end{pmatrix}, p(1) \right] \right\} \quad (4)$$

When trained with the above data set the NNs learn the value of the parameters at the given time instants. The use of this method is that when the value of the

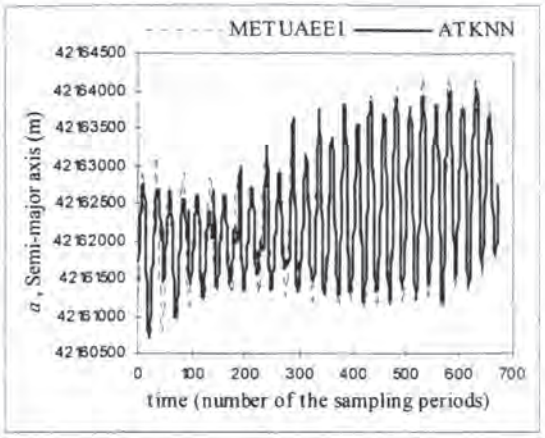

Figure 3-16 Variation of the semi-major axis in 14 days calculated by the METUAEE1 and estimated by the ATKNN with the off-line training method

parameter at a specific time is to be estimated, one can directly give that specific time instant to the NN then and get an estimation for that instant. Once the training is complete, the NNs provide the orbit parameter curves as functions of time.

Figure 3-16 shows the performance of the ATKNN in estimating the semi-major axis in the trained time interval. The training data set contains the semi-major axis values for a two weeks interval with 30 minutes sampling period. The NN with one hidden layer having 50 neurons is trained for 10,000 iterations. After training the evolution of the semi-major axis in the following 7 days is estimated and the estimation capability of the ATKNN is tested. In the on-line training method the NNs were trained and then the parameter in the next sampling point was estimated. Only the estimation error of the NNs were considered to determine its performance. How well

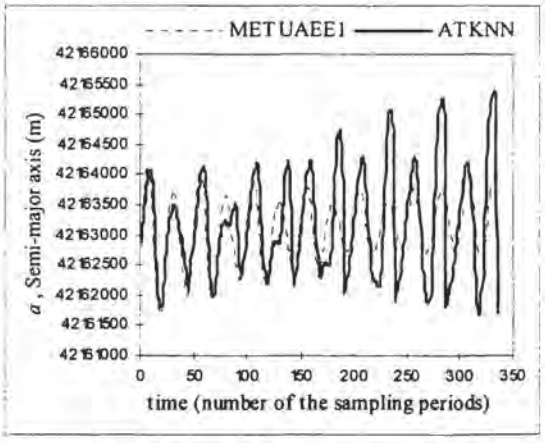

Figure 3-17 Variation of the semi-major axis in 7 days calculated by the METUAEE1 and estimated by the ATKNN with the off-line training method

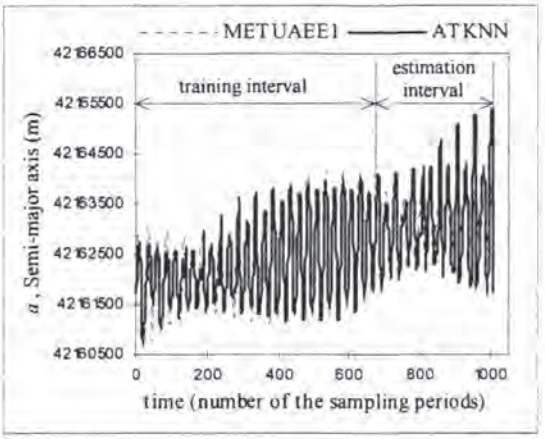

Figure 3-18 Variation of the semi-major axis in the training and estimation intervals

the training data set was learnt was not taken into account while determining the performance of the ATKNN. In the off-line training method however, both the training error and the estimation error are considered.

Figure 3-17 shows the ATKNN's estimation of the semi-major axis in the following 7 days again as compared to the METUAEE1's calculations. Figure 3-16 shows the performance of the ATKNN in learning the training data set where Figure 3-17 shows its performance in estimating the parameter in the interval it has never seen before. The estimation interval of 7 days is coming just after the training interval of 14 days. Figure 3-18 shows the ATKNN's estimation of the semi-major axis in the combined training and estimation intervals. Actually this curve is the addition of the Figure 3-17 to the end of the Figure 3-16.

We intuitively expect that as the estimation time gets farther from the training interval the error in the

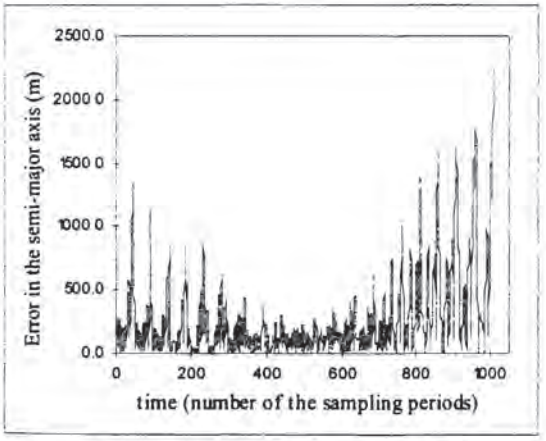

Figure 3-19 Difference between the estimations of the ATKNN and the calculations of the METUAEE1 in the 21 days period

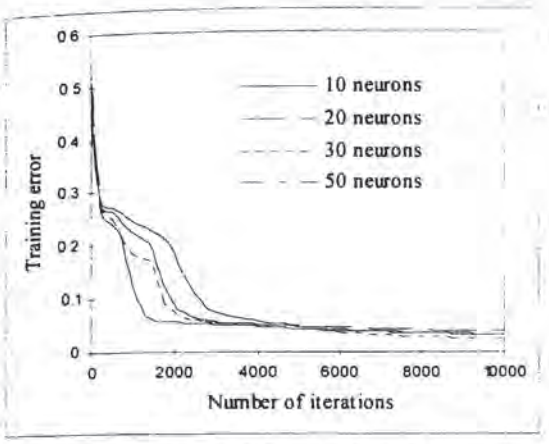

Figure 3-20 Effect of the number of iterations and number of neurons on the training error

estimation increases. Figure 3-19 shows the difference between the estimations of the ATKNN and the calculations of the METUAEE1 in the 21 days period covering the training and the estimation intervals. This figure verifies our intuitions. The first 672 sampling periods constitute the training interval and the following 336 sampling periods constitute the estimation interval. As is it is seen from this figure the error increases nonlinearly to the end of the estimation interval.

The normalized rms error at the end of 10,000 iterations is $5.89*10^{-6}$ for the training and $15.89*10^{-6}$ for the estimation. The estimation error is greater than the training error as expected. The estimation error achieved is approximately 5 times greater than that obtained in the on-line training method. In the on-line training method the semi-major axis could be estimated 90 minutes in advance with an error of $3.25*10^{-6}$. In the off-line training method the error in the estimation of the semi-major axis 7 days in advance is $15.89*10^{-6}$. That is, the estimation range

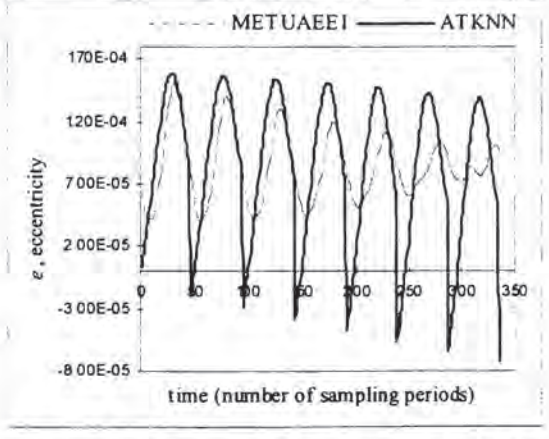

Figure 3-21 Variation of the eccentricity in 7 days calculated by the METUAEE1 and estimated by the ATKNN with the off-line training method

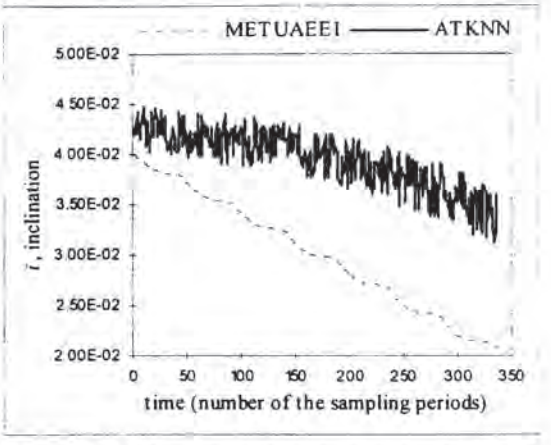

Figure 3-22 Variation of the inclination in 7 days calculated by the METUAEE1 and estimated by the ATKNN with the off-line training method

is 112 times greater than that of the previous method while the error is increased only with a factor of 5.

As it was done in the on-line training method, to determine the number of neurons and number of iterations, an optimization study is performed. Figure 3-20 shows the variation of the training error with the number of iterations and the number of neurons. Using the information given in this figure, number of neurons to be used is selected as 10 and number of iterations is selected as 2000.

Figure 3-21 and Figure 3-22 show the estimation of the eccentricity (e) and the inclination (i) respectively in the 7 days period coming after the training period of 14 days. The ATKNN's estimation of the inclination looks much different from the actual data. The reason for this becomes clear when Figure 3-23 is examined. This figure shows the variation of the inclination both in the training and the estimation intervals. As it is seen from this figure the inclination almost draws a line in the interval used for training. Then it deviates from the line in the estimation interval. Since the training interval does not cover such a variation the ATKNN could not catch the actual curve in the estimation interval.

	Normalized training error	Normalized estimation error
Eccentricity	0.2348	0.6577
Inclination	0.0297	0.3860

Table 3-5 Normalized training and estimation errors in the eccentricity and inclination

The normalized training and estimation errors for these simulations are presented in Table 3-5. The estimation errors of the off-line training method are much greater than those of the on-line training

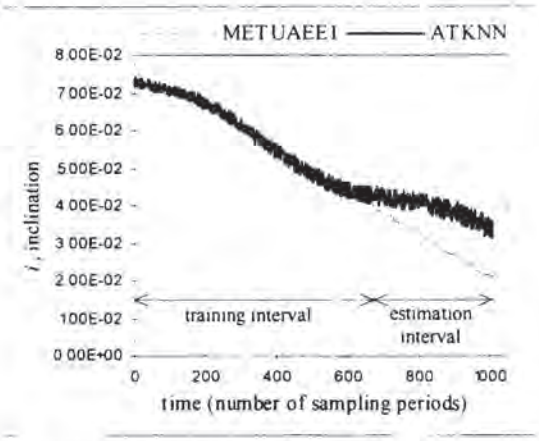

Figure 3-23 Variation of the inclination in the training and estimation intervals

method. However, it should be noted that in the on-line training method the parameters could be estimated only a few hours in advance. In the off-line training method, the estimation range is increased to the order of weeks by sacrificing the accuracy.

4. Discussion

Orbit prediction of a geostationary satellite can be performed by employing a Neural Network (NN) model. A computer program in FORTRAN 77 is developed (ATKNN) to simulate a general Multi Layer Perceptron (MLP) type NN model. The parameters that define the structure of the NN, the number of inputs, the number of layers, the number of neurons could be adjusted manually in order to obtain the best performance from the ATKNN. Two methods to estimate the orbital parameters are developed different from each other in the training procedures. In the first method the ATKNN is trained at each sampling interval during the simulation and thus referred as the "on-line training method". In the second method the ATKNN is trained before the simulation and during the simulation no more training is performed and the orbital parameters are estimated with the NNs that were trained before the simulation. So, this method is referred as the "off-line training method". Steepest Descent algorithm with one-dimensional search is employed to train the NNs. The rms and the normalized rms errors are used to measure the performance of the ATKNN. To choose the parameters to set the structure of the NNs (the number of layers and the number of neurons) many simulations are performed and the effect of each parameter on the performance of the NN is observed. The values that gave the best performance are used for these parameters. Simulations are performed using the data of the TÜRKSAT 1B satellite. The results are presented and comments are made on the results obtained. On-line training method gave much better results for the near future estimations. If the estimations for later instants are required, the off-line training method appeared as the superior method.

In all of the simulations performed in this work the synthetic data of the TÜRKSAT 1B satellite generated by the METUAEE1 code was used. With the use of the real orbital data of a spacecraft more realistic results could be obtained.

5. References

[1] C. Sakaci, Two Simulation Models: Low Altitude Flows and the TURKSAT Satellite Orbit, M. S. Thesis, Middle East Technical University Aeronautical Engineering Department, Turkey, 1996.

[2] D. Brouver, G. M. Clemence, Methods of Celestial Mechanics, Academic Press, London, 1961.

[3] A. E. Roy, Orbital Motion, Institute of Physics Publishing, Bristol and Philadelphia, 1994.

[4] P. R. Escobal, Methods of Orbit Determination, John Wiley & Sons, New York, 1965.

[5] G. Tokdemir, T. Erdal, A. T. Kutay, T. Voyvodaoğlu, Y. Tulunay, O. Tekinalp, Maneuver Simulation on the TÜRKSAT 1B Satellite Simulator, Middle East Technical University Aeronautical Engineering Department Control and Avionics Laboratory Report, Turkey, 1996.

[6] A. Cichocki, R. Unbehauen, Neural Networks for Optimization and Signal Processing, John Wiley & Sons, 1992

[7] D. G. Luenberger, Linear and Nonlinear Programming, Addison-Wesley, 1984