CHAPTER IV

EXPERIMENTAL RESULTS

4.1 Result of Proposing A New Behavior-Based Cost Function

4.1.1 Approximation problem

From the algorithm that has been tested with the standard benchmark data sets whereby, each testing set is composed of input and target values. There are both artificial and real data in our testing set.

The first testing set: the sine cardinal function, also called the *sinc* function, is the classical standard one dimensional function. The *sinc* is a function that arises frequently in signal processing and the theory of Fourier transform, defined as follows

$$f(x) = \sin c\,|x| = \frac{\sin |x|}{x} \tag{4.1}$$

This testing set is considered on the basis of a sequence of 128 measurements without noise on the uniform lattices [23].

The second testing set: The two dimensional *sinc* function contains the basis of a sequence of 1681 measurements without noise on the uniform lattices [23], as shown in the following equation.

$$f(x_1, x_2) = \sin c = \sqrt{x_1^2 + x_2^2} \tag{4.2}$$

The third and the forth testing sets are the well known chaotic time series, the Mackey-Glass[24]. This function is described by the delay-differential equation. The Δ parameters equals to 17 and 30. These two time series are denoted by MG_{17} and MG_{30}.

$$\frac{dx(t)}{t} = -0.1x(t) + \frac{0.2x(t-\Delta)}{1 + x(t-\Delta)^{10}} \tag{4.3}$$

For the fourth testing set, the Lorenz differential equations[25, 26] are used as a time series data, as follows.

$$\begin{aligned} \frac{dx}{dt} &= -c(x-y) \\ \frac{dy}{dt} &= ax - y - xz \\ \frac{dz}{dt} &= xy - bz \end{aligned}$$

where, a, b and c are constants.

There are two real world testing sets used in this experiment, the Titanium[27] and the Sunspot[28, 29] series.

Test set name	Hidden Unit Number	ϵ	Number of Training Patterns	Number of Testing Patterns
Sinc_1d	20	1.0e-4	85	43
Sinc_2d	27	1.0e-4	1120	561
MG17	19	1.5e-5	131	66
MG30	5	1.0e-3	131	66
Lorenz	40	1.0e-8	198	99
Titanium	6	1.5e-4	30	16
sunspot	21	1.0e-2	64	33

Table 4.1: The properties of the experimental data.

Table 4.1 shows the properties of the experimental data. The original data sets are separated into training and testing sets in the 2 : 1 ratio. The criterion is the first pattern belongs to the testing data, the second and the third patterns belong to the training set, consequently. Column 1 shows testing set names. Column 2 shows hidden unit numbers. Column 3 shows acceptable error (ϵ). Column 4 shows the number of training patterns and column 5 shows the number of testing patterns.

The performance of this technique is evaluated by considering the target value and the generated output value as vectors and measuring the values of *cosine* between these two vectors. Table 4.2 shows the result values from all testing sets. Columns 3 to 5 show the results from our proposed cost function, comparing with the result from column 6 to 8, which using standard LM method. Column 1 shows testing set names (Test). Column 2 shows learning rate value (η) from equation 3.12. Column 3 shows number of epoch. Column 4 shows cosine value between target and output in training phase (Cos (train)). Column 5 shows cosine value between target and output in testing phase. Column 6 shows number of epoch training by the LM algorithm, using the same initial weights as the proposed method (Cos (test)). Column 7 shows cosine value between target and output in training phase using standard LM (CosLM (train)). Column 8 shows cosine value between target and output in testing phase using standard LM (CosLM (test)). Column 9 shows t-test value between output from our proposed method and standard LM method.

For the fourth testing set, the Lorenz differential equations[25, 26] are used as a time series data, as follows.

$$\begin{aligned}\frac{dx}{dt} &= -c(x-y)\\ \frac{dy}{dt} &= ax - y - xz\\ \frac{dz}{dt} &= xy - bz\end{aligned}$$

where, a, b and c are constants.

There are two real world testing sets used in this experiment, the Titanium[27] and the Sunspot[28, 29] series.

Test set name	Hidden Unit Number	ϵ	Number of Training Patterns	Number of Testing Patterns
Sinc_1d	20	1.0e-4	85	43
Sinc_2d	27	1.0e-4	1120	561
MG17	19	1.5e-5	131	66
MG30	5	1.0e-3	131	66
Lorenz	40	1.0e-8	198	99
Titanium	6	1.5e-4	30	16
sunspot	21	1.0e-2	64	33

Table 4.1: The properties of the experimental data.

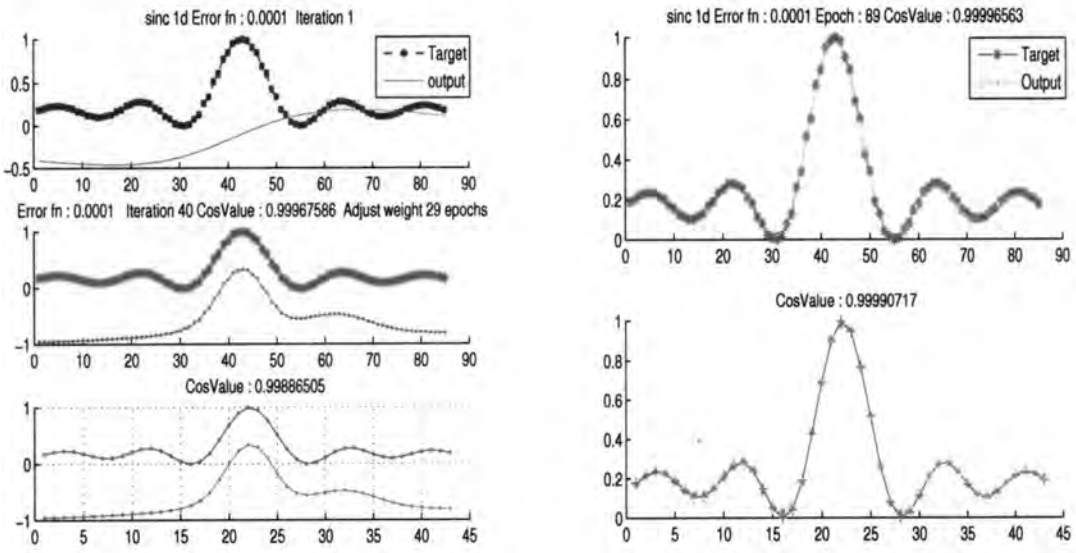
Table 4.1 shows the properties of the experimental data. The original data sets are separated into training and testing sets in the 2 : 1 ratio. The criterion is the first pattern belongs to the testing data, the second and the third patterns belong to the training set, consequently. Column 1 shows testing set names. Column 2 shows hidden unit numbers. Column 3 shows acceptable error (ϵ). Column 4 shows the number of training patterns and column 5 shows the number of testing patterns.

The performance of this technique is evaluated by considering the target value and the generated output value as vectors and measuring the values of *cosine* between these two vectors. Table 4.2 shows the result values from all testing sets. Columns 3 to 5 show the results from our proposed cost function, comparing with the result from column 6 to 8, which using standard LM method. Column 1 shows testing set names (Test). Column 2 shows learning rate value (η) from equation 3.12. Column 3 shows number of epoch. Column 4 shows cosine value between target and output in training phase (Cos (train)). Column 5 shows cosine value between target and output in testing phase. Column 6 shows number of epoch training by the LM algorithm, using the same initial weights as the proposed method (Cos (test)). Column 7 shows cosine value between target and output in training phase using standard LM (CosLM (train)). Column 8 shows cosine value between target and output in testing phase using standard LM (CosLM (test)). Column 9 shows t-test value between output from our proposed method and standard LM method.

Test	η	Epoch	Cos (train)	Cos (test)	Epoch	CosLM (train)	CosLM (test)	T-test
Sinc_1d	0.8	40	0.9996	0.9988	89	0.9999	0.9999	0.3595
Sinc_2d	0.8	102	0.9993	0.9984	507	0.9990	0.9990	0.0134
MG17	0.5	338	0.9997	0.9978	5000	0.9999	0.9914	0.0134
MG30	1.0	198	0.9934	0.9817	2000	0.9910	0.9785	0.0056
Lorenz	0.1	76	0.9999	0.9999	2104	0.9999	0.9997	0.0460
Titanium	1.0	12	0.9993	0.9997	17	0.9999	0.9991	0.0965
sunspot	1.0	11	0.9923	0.9872	12	0.9937	0.9880	0.0013

Table 4.2: The comparison of the results in the approximation problem (number of epochs and cosine values) obtained from our paper and standard LM method.

From table 4.2, the proposed cost function can forecast the output in the same direction. Less epoch were used because wherever initial weight locates, network adjusts weight to parallel. Moreover, α value setting helps increasing weight adjusting area. The *cosine* value from our proposed method and standard LM method are not much different. Figure 4.1(a) shows the result of *sinc* testing

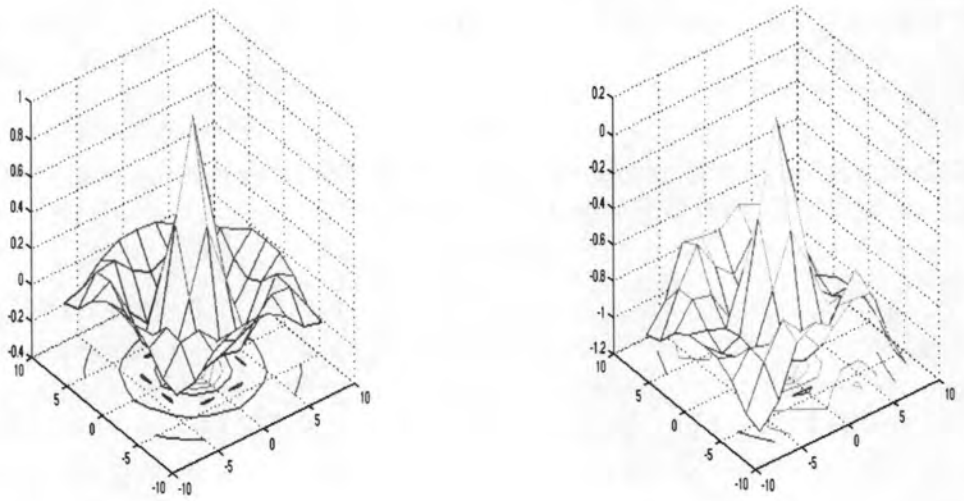


(a) (b)

Figure 4.1: This figure shows the results from the *sinc* testing set.

set obtained from the new cost function. The top figure illustrates the output of the initial weight. Dot line is the target vector. Line is the output vector. The middle figure illustrates the output of the trained network with the training time of 40 epochs. The acceptable error (ϵ) is 1.0e-4. Cosine between target and output is 0.99967586. The bottom figure illustrates the output of the testing phase. Cosine between target and output is 0.9986505.

Figure 4.1(b) shows the result of *sinc* testing set obtained from standard LM method. The top figure illustrates the output of the trained network. Dotted line is the target vector. Line is the output vector. The bottom figure illustrates the output of the testing phase.



(a) (b)

Figure 4.2: The comparison between target and output from the proposed cost function of testing set 2, two dimensional sinc: (a) The target image (b)The output image.

Figure 4.2 shows the result from the two dimensional *sinc* testing set, plotting in the two dimensional space. Figure 4.2 (a) shows the target image. Figure 4.2(b) shows the output image.

The comparison of the target results and the output results has shown in Figure 4.3. The top graph represents the target and bottom graph represent the output.

Similar to Figure 4.1, Figure 4.4 , Figure 4.5, Figure 4.6, Figure 4.7, Figure 4.8 show the result from the MG_{17}, MG_{30}, Lorenz, Titanium and Sunspot testing set, respectively.

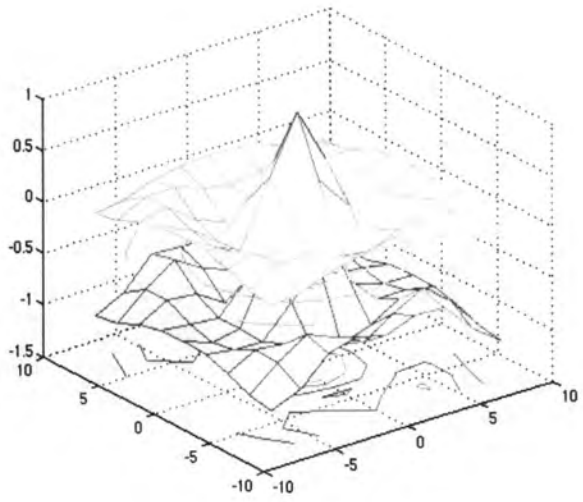
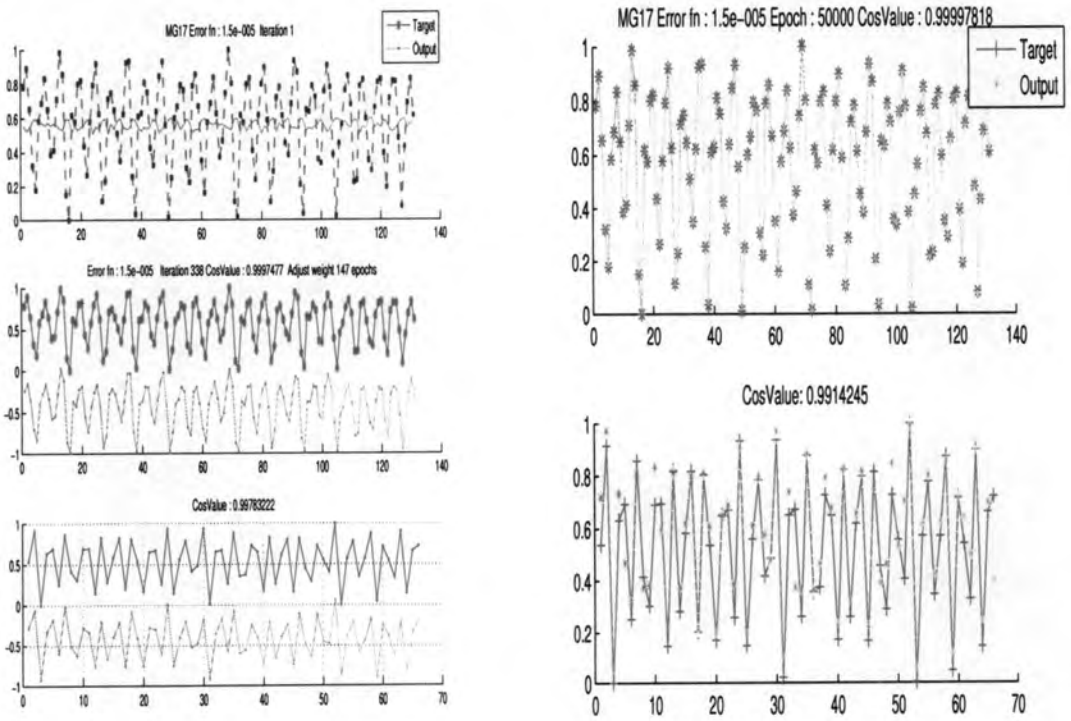
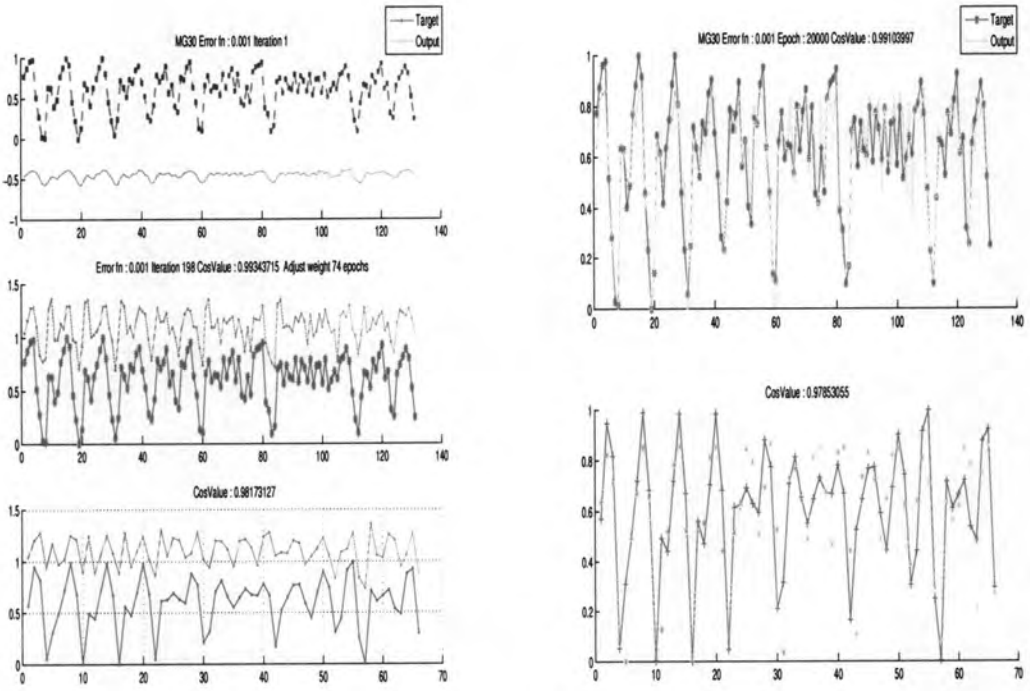


Figure 4.3: The comparison between target and output of testing set 2, two dimensional sinc. The top graph represents the target and bottom graph represent the output.



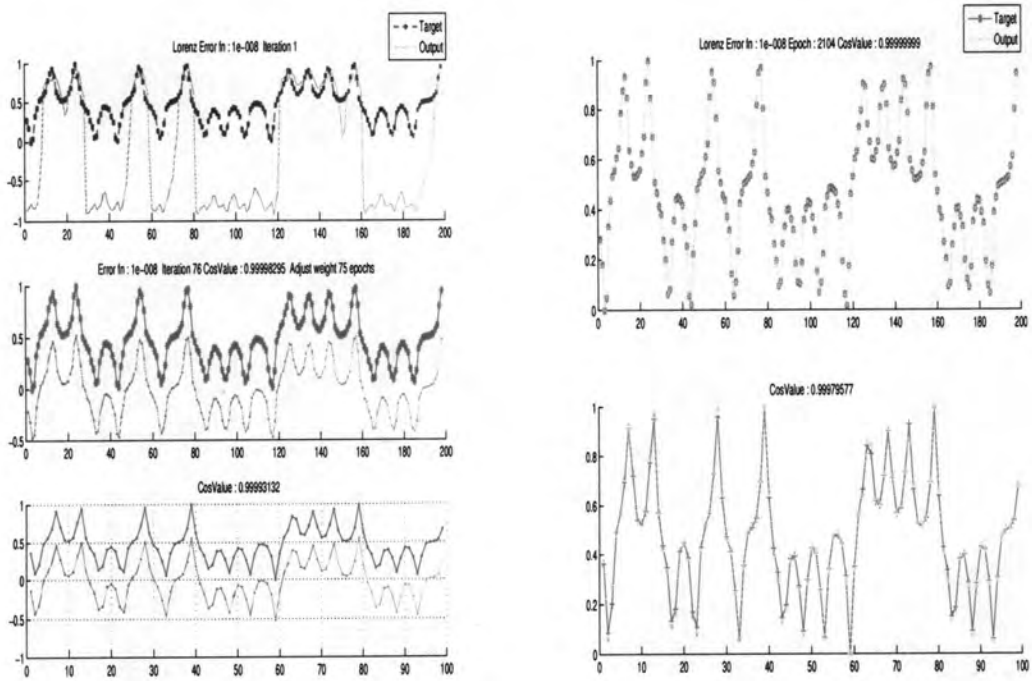
(a) (b)

Figure 4.4: Result from MG_{17} testing set.



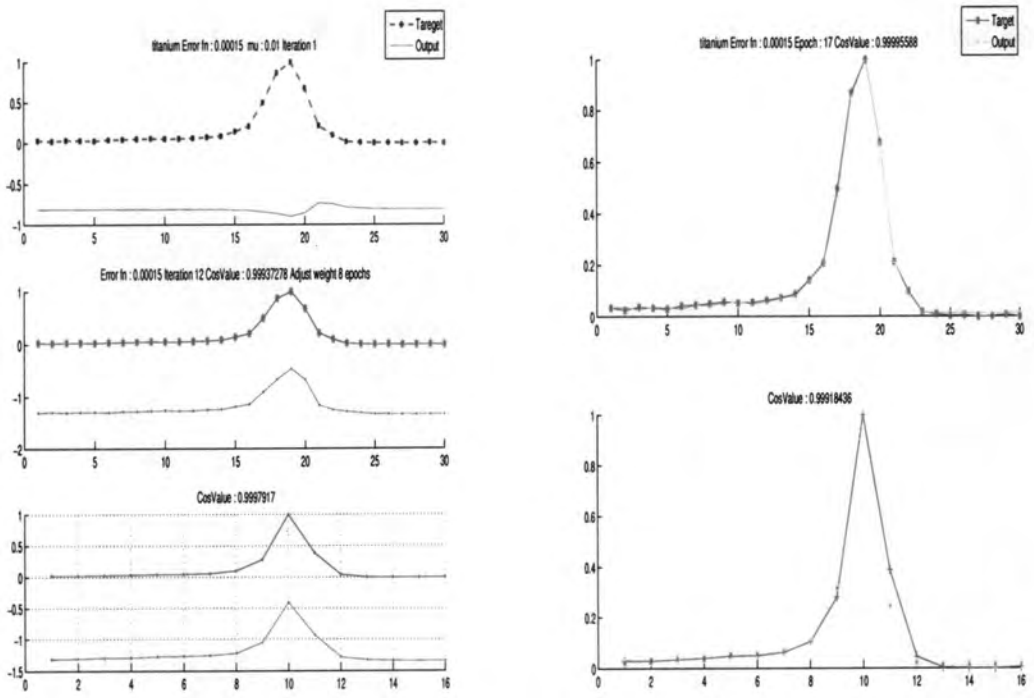
(a) (b)

Figure 4.5: Result from MG_{30} testing set.



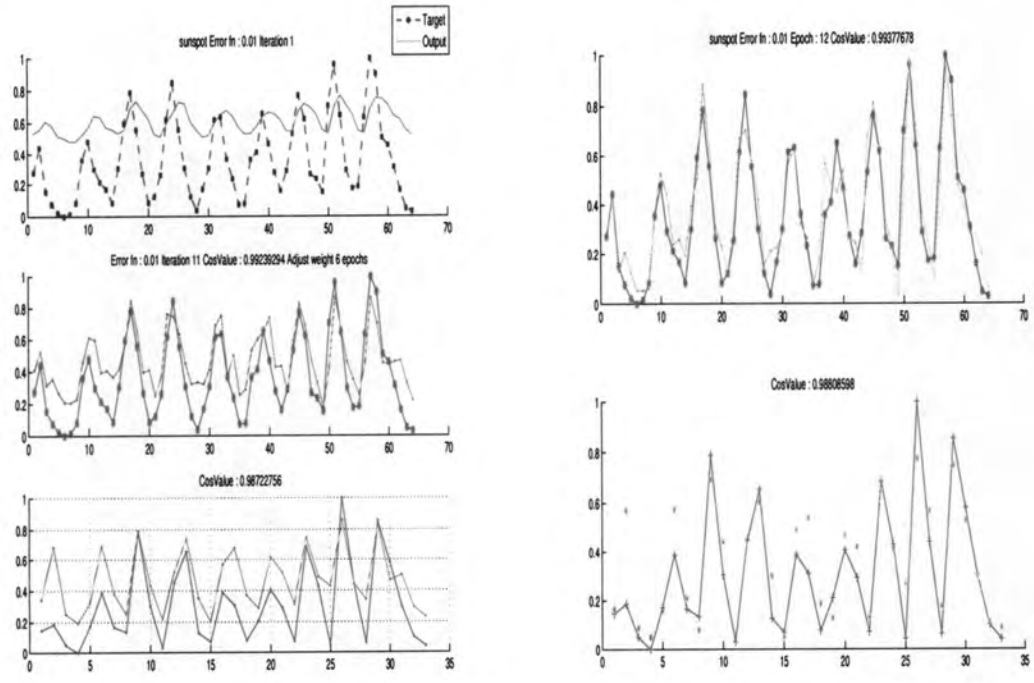
(a) (b)

Figure 4.6: Result from lorenz testing set.



(a) (b)

Figure 4.7: Result from titanium testing set.



(a) (b)

Figure 4.8: Result from sunspot testing set.

4.1.2 Classification problem

Beside the approximation problem, the proposed cost function is applied to classification problem. There are two classification testing sets. First is *xor* problem, the basic classification problem, divided into two classes, and four patterns. Second is the benchmark, called *iris* problem. It is composed of three classes, and 150 patterns. These patterns are divided into 100 train patterns and 50 test patterns.

Input pattern sequences

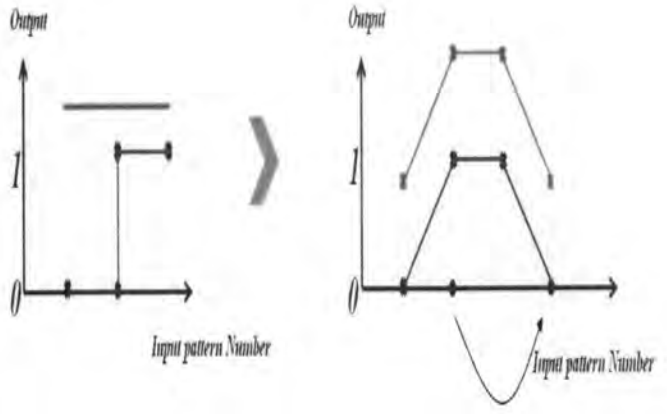


Figure 4.9: This figure shows the concept of input pattern sequence in *xor* testing set.

To learn all the data range both maximum and minimum values, input pattern sequence is very important. Figure 4.9 shows the concept of input pattern sequence. Exclusive or (*xor*) separates data into two classes and consists of four input patterns. The red line of the left figure shows target vector. Based on the concept of parallel behavior, the blue line shows output vector from our purpose function. Although, most of output vectors are paralleled with the target vectors, the output vectors cannot separate data into two classes. Therefore, we need the way to manage the input pattern sequence. The input sequence concept will help the network to create the different slope from the consecutive pattern. In the right figure, the second input pattern is transfered to be the last input pattern. Therefore, the network can learn the parallel behavior as shown in the blue line.

Figure 4.10 shows the concept of input pattern sequence in the general case. From the top image in Figure 4.10, there are 12 input patterns that separate

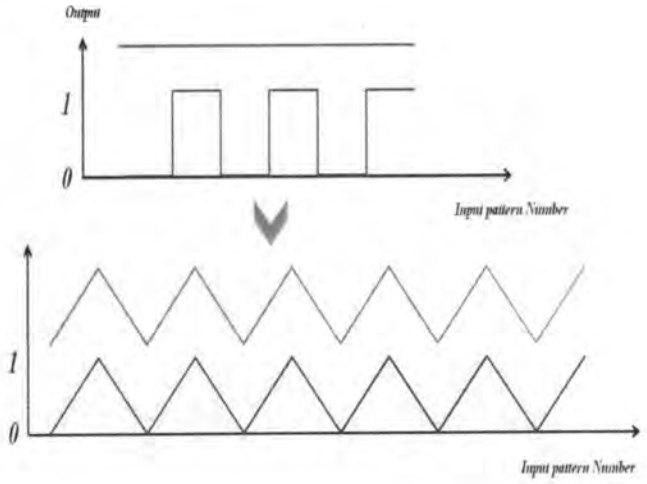


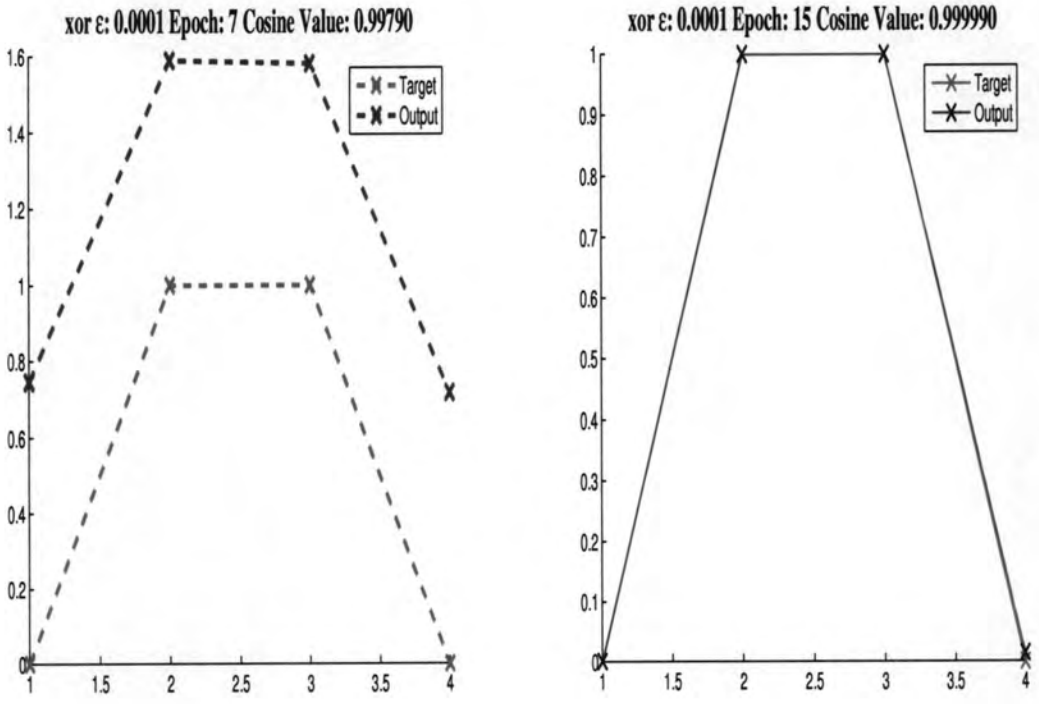
Figure 4.10: This figure shows the concept of input pattern sequence in the general case.

into two classes. To avoiding output that receiving from the cost function which display as a blue line, the bellow image shows the idea of input pattern sequence. The input pattern sequence concept is the consequent pattern that must come form the different classes. Doing in the same way for every training pattern, it will help the network to learn the different slope from the consequent pattern.

Figure 4.11(a) shows the result of *xor* testing set obtained from the proposed cost function. The blue line illustrates the output value. The red line is the target vector. Figure 4.11(b) shows the result of *xor* testing set obtained from standard LM method.

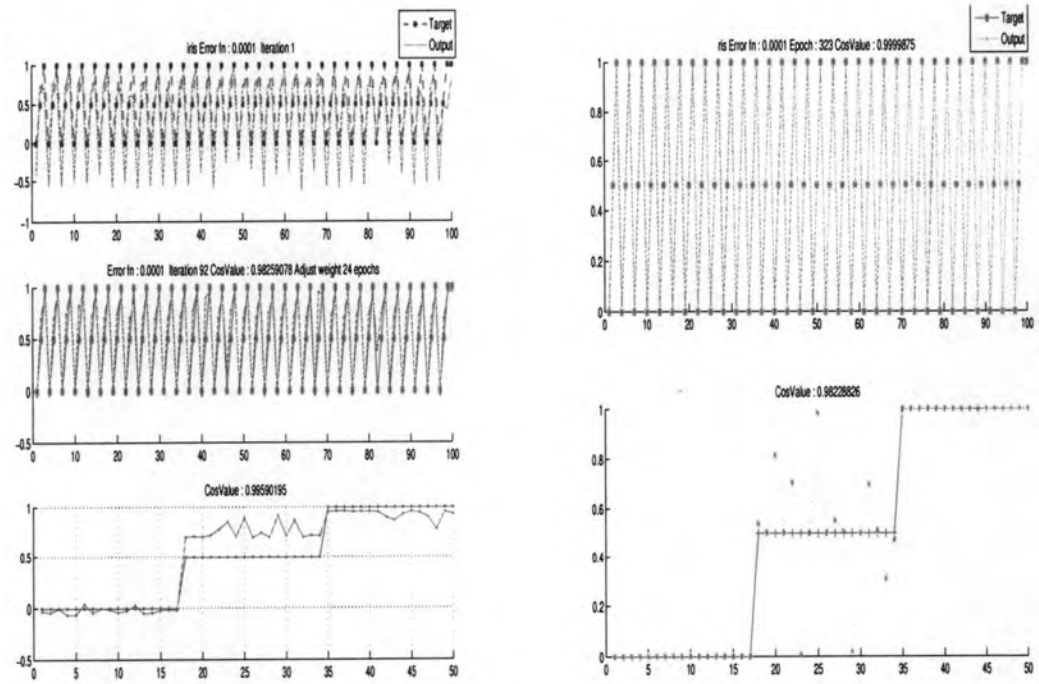
Therefore, to accelerate and cover the learning pattern in *iris* testing set, we fed the input pattern by consecutive classes. The *iris* problem has three classes. The first input pattern is from class one, the second is from class two, the third is from class three, the fourth is from class one, and so on. Figure 4.12(a) shows the result of *iris* testing set obtained from the new cost function. The top figure illustrates the output from the initial weight. Dot line is the target vector. Line is the output vector. The middle figure illustrates the output of the trained network, and the bottom figure illustrates the output of the testing phase.

Figure 4.12(b) shows the result of *iris* testing set obtained from standard LM method. The top figure illustrates the output of the trained network. Dotted line is the target vector. Line is the output vector. The bottom figure illustrates the output of the testing phase.



(a) (b)

Figure 4.11: Result from xor testing set



(a) (b)

Figure 4.12: Result from iris testing set.

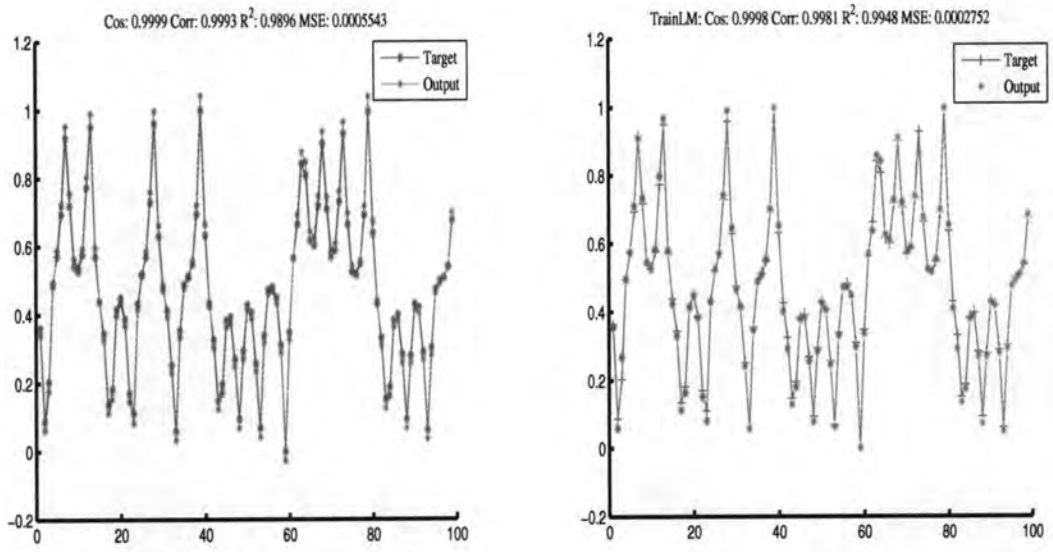
Test	Hidden UnitNo	ϵ	η	Epoch	Cos (train)	Cos (test)	Epoch	CosLM (train)	CosLM (test)
xor	2	1.0e-4	1	7	0.99790	0.99790	15	0.99999	0.99999
iris	45	1.0e-4	1	92	0.98259	0.99590	323	0.99998	0.98228

Table 4.3: The comparison of the results of the classification problem (number of hidden units, number of epochs and cosine value) obtained from our method and from normal LM method.

Similar to Table 4.2, Table 4.3 shows the result values from the classification problem. Columns 4 to 6 show the results from our proposed cost function, comparing with the result from column 4 to 9, which using standard LM method.

4.2 Shifting the Result

In this section, we apply the shifting result technique with some testing sets of the approximation problem. Using equation 3.22, we compute the $shift$ value. Figure 4.13 shows the shifting technique result of the Lorenz testing set comparing with the result from standard LM.



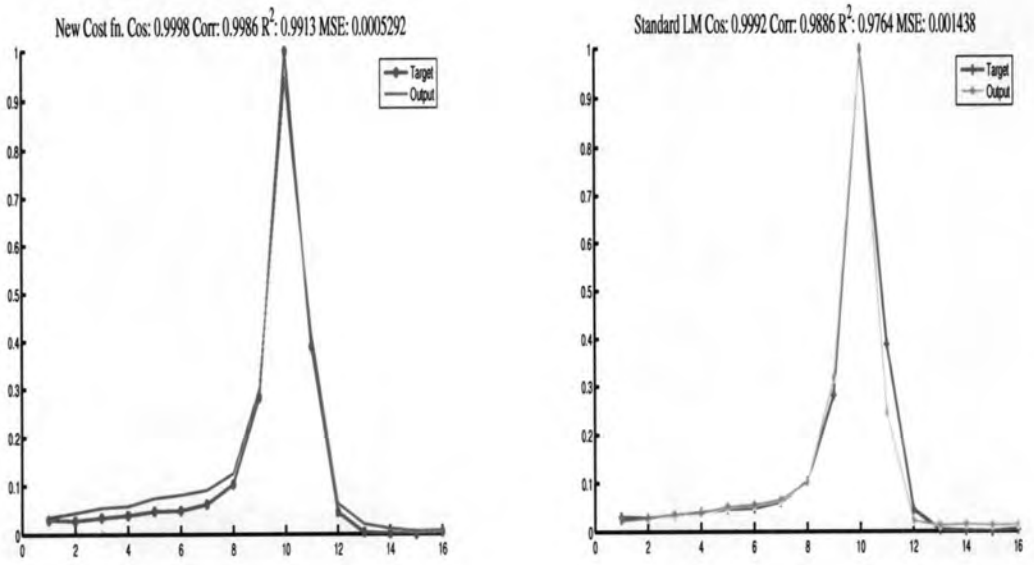
(a) (b)

Figure 4.13: Shifting the result of Lorenz testing set with the $shift$ value = 0.4827.

Figure 4.13 (a) shows the result from shifting output to get closer with the target. in testing phase. Figure 4.13 (b) shows the result from standard LM.

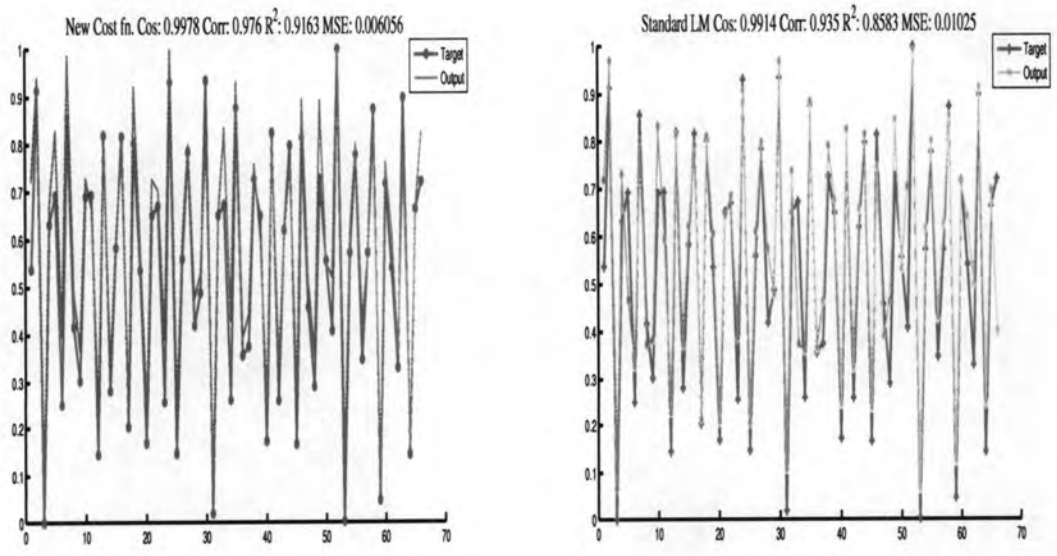
The error measurements are measured the cosine value (Cos), co-relation ($Corr$), co-efficient of determination (R^2), and means squared error (MSE), respectively.

Similar to Figure 4.13, Figure 4.14 , Figure 4.15, Figure 4.16, Figure 4.17 show the result from the Titanium, MG_{17}, MG_{30} and Sunspot testing set, respectively.



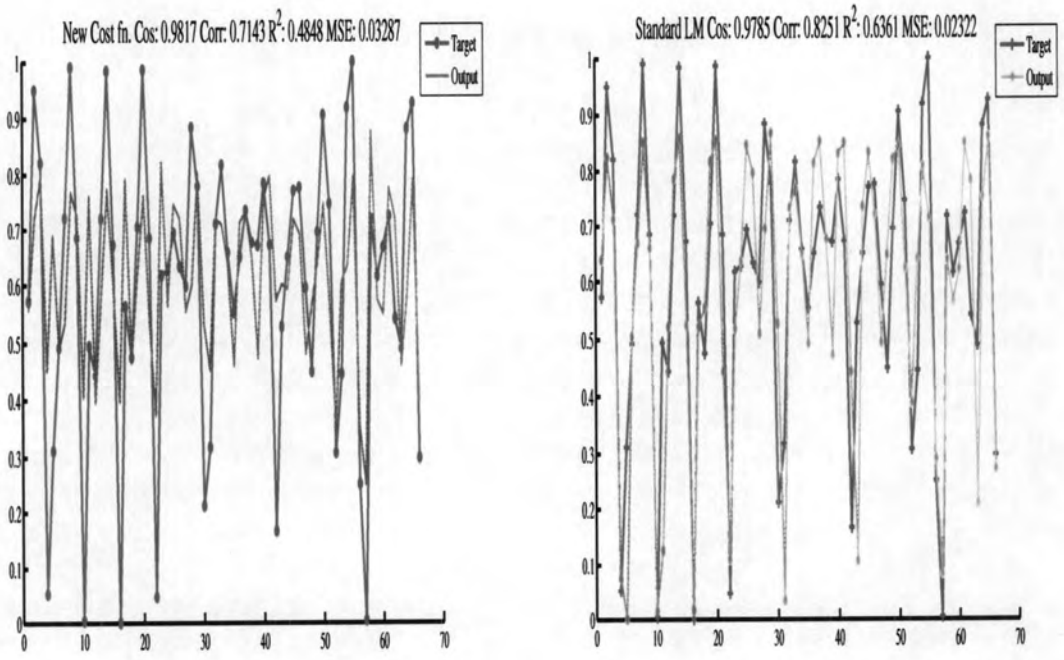
(a) (b)

Figure 4.14: Shifting the result of Titanium testing set with the $shift$ value = 1.347.



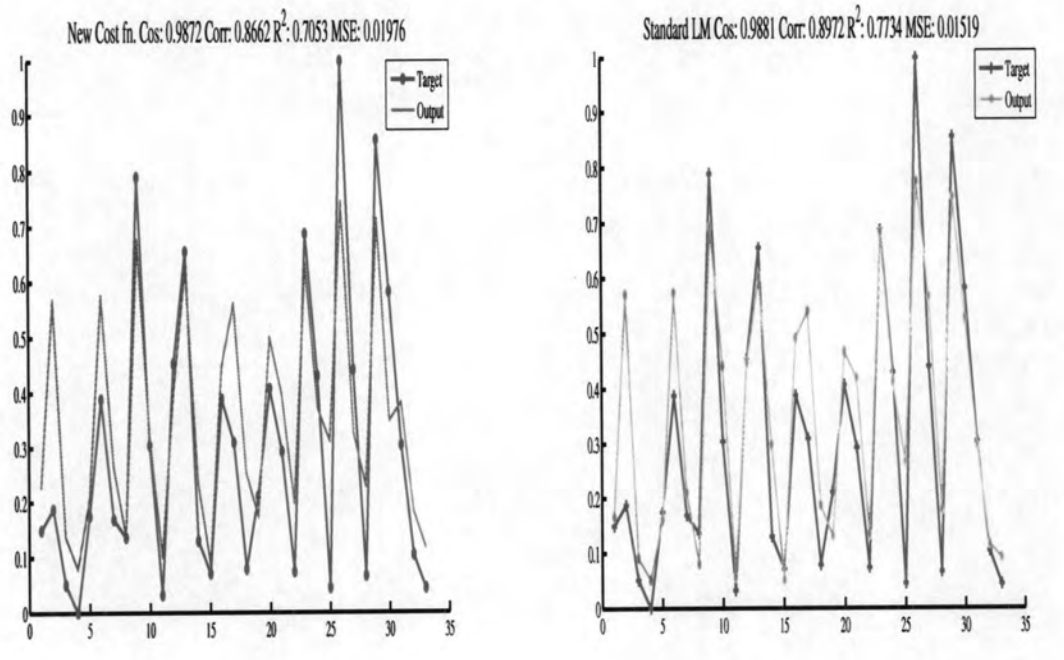
(a) (b)

Figure 4.15: Shifting the result of MG17 testing set with the $shift$ value = 0.9997.



(a) (b)

Figure 4.16: Shifting the result of MG30 testing set with the $shift$ value = -0.4891.



(a) (b)

Figure 4.17: Shifting the result of Sunspot testing set with the $shift$ value = -0.1168.

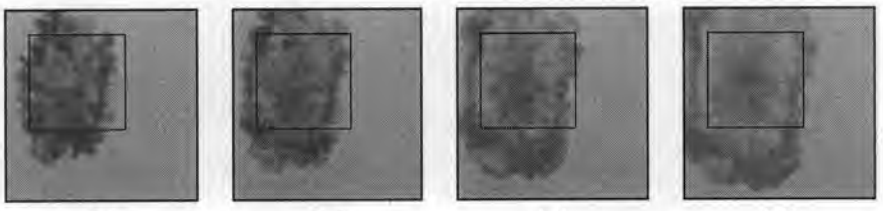
4.3 Result of Using Neural Cellular Automata To Simulate The Color Diffusion

This experiment composes of two test sets. The first test set perform by drop colored liquid onto water surface. The second test set inject the colored liquid on water surface with blocking object.

For the test set 1, there are two kinds of test sets which is using nine input for training (3×3) and using 25 inputs for training (5×5). The training of 3 BPNN uses nine input units, 20 hidden units and one output unit. Its learning rate is equal to 0.9, and training time is equal to 12 epochs. and the training of 5×5 BPNN of uses 25 input units, 20 hidden units and one output unit. Its learning rate is equal to 0.9, and training time is equal to 46 epochs.

Figure 4.18 shows the actual target images obtained from the colored liquid dropping experiment and Figure 4.19 shows the output images predicted by neural network. The value of each pixel in this network is computed by the values from its square neighboring pixels of size 3×3. The area within the rectangle refers to the trained area, and the one outside the rectangle refers to the untrained area. The reason of introducing untrained area to neural network is to prove that neuron network is not only capable of predicting the trained area beyond the training time, but also the untrained one.

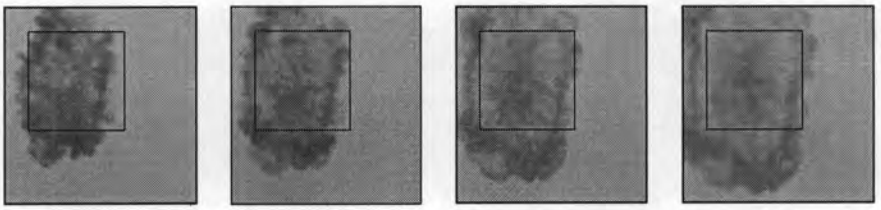
The experiment by using a square neighboring of size 5×5 is also conducted. The result is shown in Figure 4.20.



(a) (b) (c) (d)

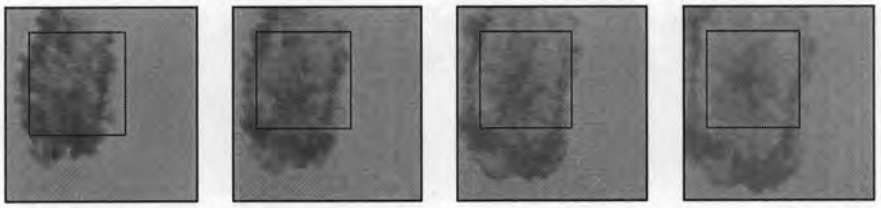
Figure 4.18: The actual snapshot images taken at different times from the experiments. The area inside the rectangle is the training area and the area outside is the testing area. (a) At time 2. (b) At time 10. (c) At time 20. (d) At time 30.

The performance of this technique is evaluated by considering the actual image and the generated image as vectors and measuring the values of *cosine* between these two vectors. Figure 4.21 shows the relationship between target and output vectors by representing their values of $cos\theta$ at different times. The value of *cosine* obtained from the neural network by the period of predictable time from time 101 to time 200. The graph shows two plots. The line refers to the predicted images of 3×3, and the dotted line refers to the predicted images of 5×5.



(a) (b) (c) (d)

Figure 4.19: The predicted images 3×3 compared with the actual image in Figure 4.18. (a) At time 2. (b) At time 10. (c) At time 20. (d) At time 30.



(a) (b) (c) (d)

Figure 4.20: The predicted images of 5×5 compared with the actual image in Figure 4.18. (a) At time 2. (b) At time 10. (c) At time 20. (d) At time 30.

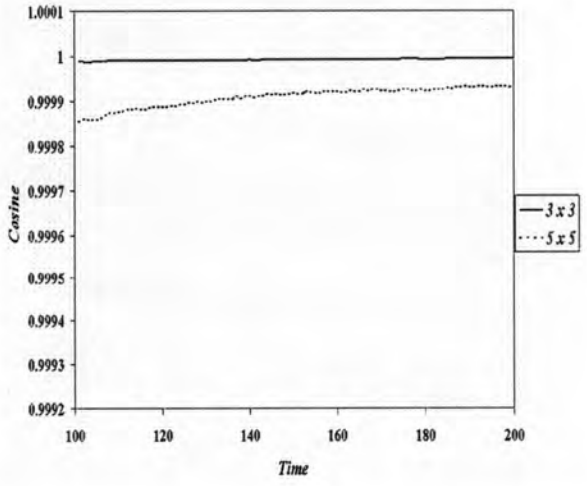


Figure 4.21: The value of *cosine* obtained from the neural network by the period of predictable time.

Furthermore, we applied the cost function to simulate and forecast the dynamical behavior of water diffusion, also gave parallel difference between the output and target values. The output value is not necessary to be the same as that of the target value. Only certain difference at every time step is acceptable.

Same as the previous result, Table 4.4 shows the result from the test set 1, applied the proposed cost function to learn the dynamical behavior of water

Test	Epoch	Cos (train)	Cos (test)	Epoch	CosLM (train)	CosLM (test)	T-test
Test set1	52	0.9897	9897	289	0.9991	0.9991	0.1563

Table 4.4: The comparison of the results in testset 1 (number of epochs and cosine values) obtained from our method and standard LM method.

diffusion. In this testing set, the learning rate equals to one and the tolerance equals to 1.0E-4. This testing set also uses 20 hidden units. The proposed cost function uses less epoch that the standard LM method because wherever initial weight locates, the network adjusts weight to parallel.

For test set 2, the resulted images are performed by injecting colored liquid onto water surface with blocking object. The trained network uses nine input units, 50 hidden units and one output unit. Its learning rate is equal to 0.7, and the tolerance equal to 0.1. The actual target images and the output images predicted by neural network are shown in Figure 4.22 and 4.23, respectively.

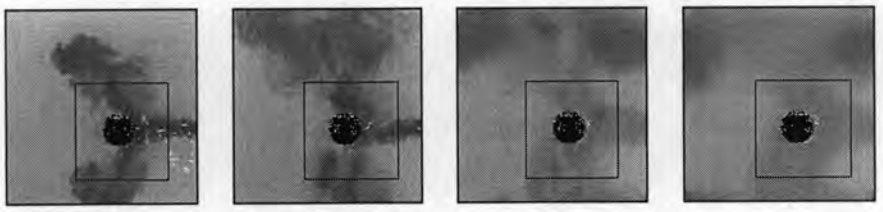


Figure 4.22: The actual snapshot images taken at different times from the test set 2. The area inside the rectangle is the training area and the area outside is the testing area. (a) At time 50. (b) At time 100. (c) At time 150. (d) At time 200.

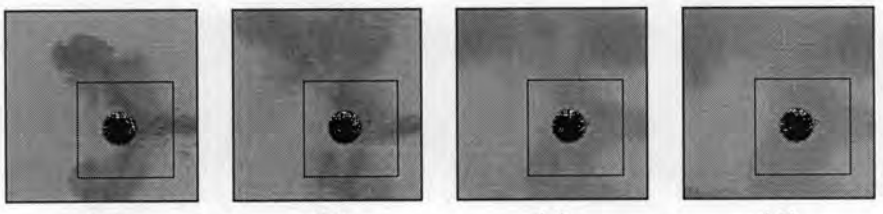


Figure 4.23: The predicted images compared with the actual image in Figure 5. (a) At time 50. (b) At time 100. (c) At time 150. (d) At time 200.

4.4 Result of Using Neural Cellular Automata To Simulate The Tsunami Behavior

We apply the idea of neural cellular automata to simulate and forecast the tsunami behavior. For improving the learning method, LM algorithm was also used as an learning algorithm to avoid the problem of nonconvergence of the training network. There are 600 sequence images in the training data. It is divided into two parts, the first 360 images for training, and the last 240 images for testing. The network use nine input units for training (3×3), 20 hidden units, and one output unit. The tolerance equals to 1.0E-4, and the training time is 1817 epochs. The value of cosine between the target and the output vector equals to 0.9635. The actual target images and the output images predicted by neural network are shown in Figure 4.24 and 4.25, respectively. The white color shows the land area, and the gray color shows the sea area.

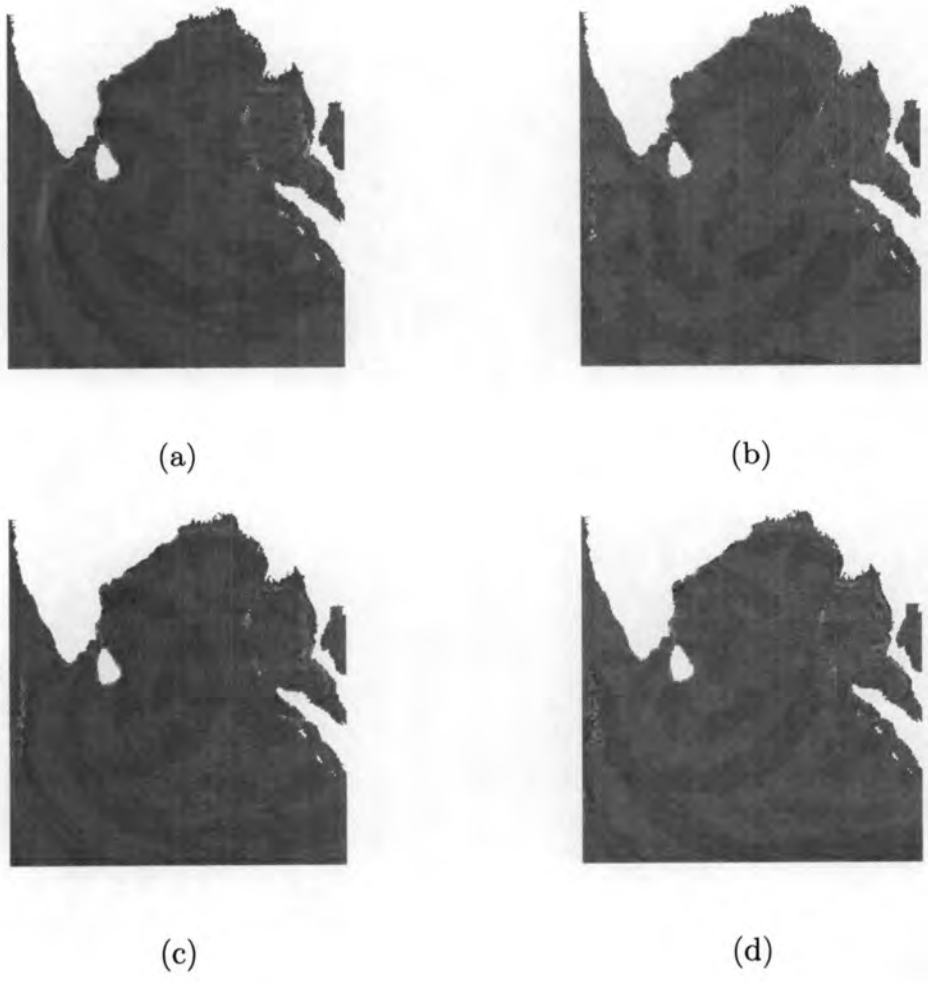
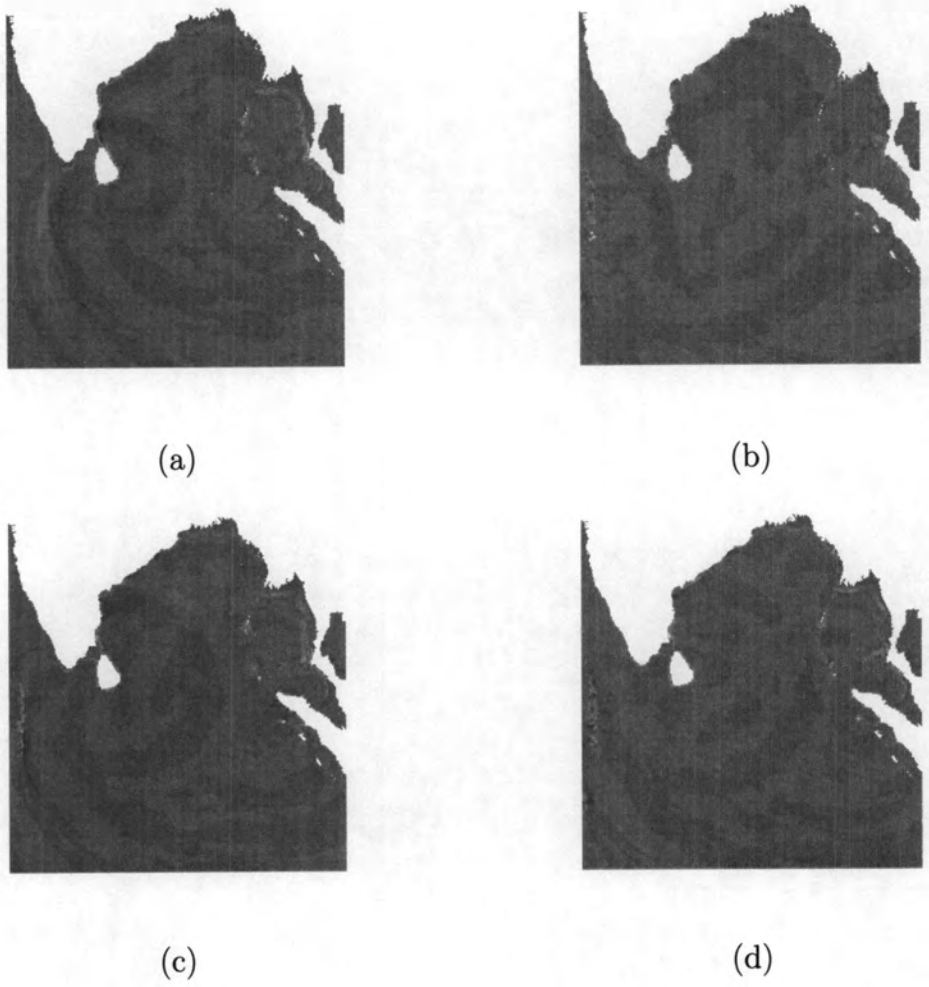


Figure 4.24: The actual snapshot images taken at different times from the experiments (a) At time 241. (b) At time 341. (c) At time 441. (d) At time 600.



(a) (b)

(c) (d)

Figure 4.25: The predicted images 3×3 compared with the actual image in Figure (a) At time 241. (b) At time 341. (c) At time 441. (d) At time 600.