International Journal of Artificial Intelligence and Applications for Smart Devices
Vol.1, No.1 (2013), pp.25-40
http://dx.doi.org/10.14257/ijaiasd.2013.1.1.03

Visualising Data Using Reservoir with Clustering

Wesam Ashour[1] and Colin Fyfe[2]

[1]*Islamic University of Gaza, Palestine*
[2]*University of the West of Scotland, UK*

[1]*washour@iugaza.edu.ps*, [2]*Colin.Fyfe@uws.ac.uk*

Abstract

In this paper we illustrate a new method of visualizing and projecting time series data using reservoir computing with clustering algorithms. We show the advantages of using clustering with reservoir to visualize data. Then we extend the clustering algorithm and use a fixed latent space to preserve the topology in the projection. We illustrate the method using airport and financial time series data.

Keywords*: visualization, echo state machines, clustering, temporal, latent space*

1. Introduction

Identifying structure in high dimensional data spaces is a difficult problem. One method frequently used is to project the data onto a low dimensional manifold and allow a human investigator to search for structure in this manifold by eye. There are many artificial neural network methods *e.g.*, [7, 10, 12, 15, 21, 22] for projecting data onto low dimensional manifolds. In this paper, we will review a visualization method based on an artificial neural network which is specifically designed to display low dimensional projections of time series data.

Time series data presents opportunities for projection which other data sets may not have: in a typical time series, nearby (in time) samples often have values which are close to one another. We will develop a neural network method which captures the dynamical nature of such data but projects the data onto a 2 dimensional manifold on which we can view the original time series. We will base the visualization property of the neural network on the neuroscale algorithm [23]. Since we wish to capture dynamical information we use a network that has a memory of the past - the echo state machine.

It is well known that real biological neural networks have many feedback connections and it is recognized that such recurrent nets have information processing powers that feed forward neural networks do not have. However while we have efficient algorithms for training feed forward neural networks, no efficient algorithms have existed for recurrent neural net- works. Reservoir computing is a relatively new type of artificial neural network which attempts to overcome this known difficulty in training recurrent neural networks. We will concentrate on a type of reservoir known as Echo state networks [6, 13, 18, 19, 25].

Data clustering techniques are an important aspect used in many fields such as data mining, pattern recognition and pattern classification, data compression, machine learning [5], image analysis [26], and bioinformatics. The purpose of clustering is to group data points into clusters in which the similar data points are grouped in the same cluster while dissimilar data points are in different clusters. The high quality of clustering is to obtain high intra-cluster similarity and low inter-cluster similarity.

The K-means algorithm is one of the most frequently used investigatory algorithms in data analysis. The algorithm attempts to locate K prototypes or means throughout a data set in such a way that the K prototype in some way best represents the data. It is an iterative algorithm in which K means are spread throughout the data and the data samples are allocated to the mean which is closest (often in Euclidean norm) to the sample. Then the K means are repositioned as the average of data points allocated to each mean. This continues until stable convergence is reached. The K-means algorithm is one of the first which a data analyst will use to investigate a new data set

ISSN: 2288-6710 IJAIASD

because it is algorithmically simple, relatively robust and gives 'good enough' answers over a wide variety of data sets: it will often not be the single best algorithm on any individual data set but it may be close to the optimal over a wide range of data sets. However the algorithm is known to suffer from the defect that the means or prototypes found depend on the initial values given to them at the start of the simulation: a typical program will converge to a local optimum. There are a number of heuristics in the literature which attempt to address this issue but, at heart, the fault lies in the performance function on which K-means is based.

[16] proposed a global K-means algorithm, an incremental approach to clustering that adds one cluster prototype at a time through a deterministic global search consisting of N (the data size) executions of the K-means; this algorithm can obtain equivalent or better results than the standard K-means, but it suffers from high computation cost and at the same time gives no guarantee to find the optimum.

Arthur and Vassilvitskii [1] improved the K-means algorithm by substituting the random allocation of the prototypes with a seeding technique. They give experimental results that show the advantage of this algorithm in time and accuracy.

In [2-4] we derive a family of new clustering algorithms that solve the problem of sensitivity to initial conditions in the K-means algorithm.

This paper includes an extension of work discussed in [24]: in that paper, we compared the new method with projections from principal component analysis and self-organizing maps of various varieties [10-12, 14, 15, 17]. In this paper, we use a different visualization method based on an underlying latent space similar to that developed for the generative topographic mapping [7].

2. Reservoir Clustering Model

Echo state networks (ESNs) consist of three layers of 'neurons': an input layer which is connected with random and fixed weights to the next layer which forms the reservoir. The neurons of the reservoir are connected to other neurons in the reservoir with a fixed, random, sparse matrix of weights. Typically only about 10% of the weights in the reservoir are non-zero. The reservoir is connected to the output neurons using weights which are trained using error descent. In this paper, we will leave the input to reservoir and the reservoir to reservoir weights fixed in their standard form but investigate training the output weights using a clustering algorithm.

In this section, we show how to use clustering algorithm with reservoir to visualize temporal data set. We use our clustering algorithm Inverse Exponential K-means (IEK), which is more robust to the initial parameters than K-means and EM algorithm. It is also provides better results regarding convergence to a local optimum.

2.1. Inverse Exponential K-means Algorithm (IEK)

To solve the problem of sensitivity in K-means, IEK has the following two objective functions:

$$J_K = \sum_{i=1}^{N} \min_{j=1}^{K} \| \mathbf{x}_i - \mathbf{m}_j \|^2 \qquad (1)$$

which is used for K-means and:

$$\begin{aligned} k* &= \arg\min_{k=1}^{K}(\| \mathbf{x}_i - \mathbf{m}_k \|) \\ J_2 &= \sum_{i=1}^{N} \left[\sum_{j \neq k*}^{K} \frac{1}{\| \mathbf{x}_i - \mathbf{m}_j \|} \right] \left(1 - \exp(- \| \mathbf{x}_i - \mathbf{m}_{k*} \|^3)\right) \end{aligned} \qquad (2)$$

The first objective function is important for clustering data, but it has limitation which causes dead prototypes and convergence to local optimum. The problem of this function is that each prototype responds only to data points that are closest to this prototype, and has not been affected by other data points. Thus it is sensitive to the prototypes initialization.

This limitation has been improved by adding another objective function which gives a relationship between all data points and all prototypes. This objective function deals with the prototypes that are not detected by the minimum function, and makes them responding to the whole

data points not only to their members (closest data points). This makes the algorithm more robust to the initial prototypes and avoids the convergence to the local optimum.

One of the advantages of IEK algorithm is that it has two sets of update rules. This is beneficial when all prototypes are far from the data set: such a situation may very well happen when we have a high dimensional data set since in that situation most of the volume of the space lies in a thin shell far from the centre of the data so that even initializing prototypes to data points will not guarantee that the prototypes are close to many samples. We use (1) when $\mathbf{m}_j$ is the closest to $\mathbf{x}_i$ and use (2) for the other prototypes that are not the closest to $\mathbf{x}_i$.

The objective function (2) deals with all prototypes that are not recognized by minimum function in (1). This function at minimum values tries to distribute the prototypes to fit the data and find the clusters.

2.1.1. Optimization and Implementation

To derive the IEK algorithm, we need to find the partial derivative of (1) with respect to $\mathbf{m}_k$ and the partial derivative of (2) with respect to $\mathbf{m}_j$ where $\mathbf{m}_k$ represents the closest prototype to $\mathbf{x}_i$, and $\mathbf{m}_j$ represents the other prototypes.

$$\frac{\partial J_K}{\partial \mathbf{m}_k} = \sum_{i \in V_k} -2(\mathbf{x}_i - \mathbf{m}_k) \tag{3}$$

where V_k contains the indices of data points that are closest to $\mathbf{m}_k$

$$\begin{aligned} \frac{\partial J_K}{\partial \mathbf{m}_k} &= 0 \\ \sum_{i \in V_k} -2(\mathbf{x}_i - \mathbf{m}_k) &= 0 \\ \mathbf{m}_k &= \frac{\sum_{i \in V_k} \mathbf{x}_i}{N_r} \\ \mathbf{m}_k &= \frac{\sum_{i \in V_k} \mathbf{x}_i a_{ik}}{\sum_{i \in V_k} a_{ik}} \end{aligned}$$

where N_r is the number of data points that are closest to $\mathbf{m}_k$, $a_{ik} = 1$.

Note that this constitutes only a part of the calculation of $\mathbf{m}_k$ from only the closest data points, however there is another calculation for $\mathbf{m}_k$ (using the rest of data points). This is provided from the second performance function as $\mathbf{m}_k$ might not be the closest to some data points xi, i ∈ V_j where V_j is the index of data points that are not closest to $\mathbf{m}_k$, see (8). Thus $\mathbf{m}_k$ should be calculated based on all the data points, not only the closest points as happens in K-means algorithm.

The second performance function provides new calculations for the prototypes that are not closest to data points, and distributes them in a good way to identify the clusters.

$$\begin{aligned} \frac{\partial J_2}{\partial \mathbf{m}_j} &= \sum_{i \in V_d} \frac{(\mathbf{x}_i - \mathbf{m}_j)(1 - \exp(- \| \mathbf{x}_i - \mathbf{m}_{k*} \|^3)}{\| \mathbf{x}_i - \mathbf{m}_j \|^3} \\ &= \sum_{i \in V_d} \frac{(\mathbf{x}_i - \mathbf{m}_j) c_i}{\| \mathbf{x}_i - \mathbf{m}_j \|^3} \end{aligned} \tag{4}$$

where V_d contains the indices of data points that are not closest to $\mathbf{m}_j$

$$c_i = 1 - \exp(- \| \mathbf{x}_i - \mathbf{m}_{k*} \|^3) \tag{5}$$

By assigning the partial derivative to zero and solving for $\mathbf{m}_j$ we have:

$$\begin{aligned} \frac{\partial J_2}{\partial \mathbf{m}_j} &= 0 \\ \Longleftrightarrow \sum_{i \in V_d} \frac{(\mathbf{x}_i - \mathbf{m}_j)c_i}{\| \mathbf{x}_i - \mathbf{m}_j \|^3} &= 0 \\ \Longleftrightarrow \sum_{i \in V_d} \frac{\mathbf{x}_i c_i}{\| \mathbf{x}_i - \mathbf{m}_j \|^3} &= \sum_{i \in V_d} \frac{\mathbf{m}_j c_i}{\| \mathbf{x}_i - \mathbf{m}_j \|^3} \end{aligned}$$

$$\begin{aligned} \mathbf{m}_j(t+1) &= \frac{\sum_{i \in V_d} \frac{\mathbf{x}_i c_i}{\|\mathbf{x}_i - \mathbf{m}_j(t)\|^3}}{\sum_{i \in V_d} \frac{c_i}{\|\mathbf{x}_i - \mathbf{m}_j(t)\|^3}} \\ \text{or } \mathbf{m}_j(t+1) &= \frac{\sum_{i \in V_d} \mathbf{x}_i b_{ij}}{\sum_{i \in V_d} b_{ij}} \end{aligned} \quad (6)$$

where,

$$\begin{aligned} b_{ij} &= \frac{c_i}{\| \mathbf{x}_i - \mathbf{m}_j(t) \|^3} \\ &= \frac{1 - \exp(- \| \mathbf{x}_i - \mathbf{m}_{k*} \|^3)}{\| \mathbf{x}_i - \mathbf{m}_j(t) \|^3} \end{aligned} \quad (7)$$

The new locations for all prototypes can be calculated by:

$$\mathbf{m}_r(t+1) = \frac{\sum_{i \in V_r} \mathbf{x}_i a_{ir} + \sum_{i \in V_j, j \neq r} \mathbf{x}_i b_{ir}}{\sum_{i \in V_r} a_{ir} + \sum_{i \in V_j, j \neq r} b_{ir}} \quad (8)$$

where V_r contains the indices of data points that are closest to $\mathbf{m}_r$, V_j contains the indices of all the other points and

$$a_{ir} = 1 \quad (9)$$

$$b_{ir} = \frac{1 - \exp(- \| \mathbf{x}_i - \mathbf{m}_{k*} \|^3}{\| \mathbf{x}_i - \mathbf{m}_r(t) \|^3} \quad (10)$$

The clustering algorithms provide advantages when we have clusters with different shapes and sizes in the dataset. In this type of data, most of recent works will not get good results as it based on linear distances *e.g.*, Euclidean distance. In this case, we find clustering idea could help by selecting a suitable clustering algorithm which finds clusters with nonlinear shapes *e.g.*, BIRCH clustering algorithm which based on density rather than distances in measuring the similarities and hence finding the clusters.

Clustering algorithms help us to discover the groups of data that are similar to each other which help in discovering data. However, it does not preserve the topology between clusters, *i.e.*, points in cluster i are not important to be close to points in cluster i+1 or i-1. Thus we intend to improve some of the clustering algorithms to preserve topology and try to apply them again with reservoir to project and visualize the temporal data set.

2.2. Simulation

Main steps:

* Create a reservoir with 300 units.
* Initialize fixed random weights between input data and reservoir.
* Initialize fixed random internal weights for the reservoir units.
* Feed the input data to the reservoir and calculate the reservoir units values.
* Number the prototypes and initialize them in the reservoir units space.
* Cluster the reservoir units using IEK.
* Draw the results in two dimensions, the first dimension represents the number of the prototype and the second dimension represents the index of the input data point. It is possible also to generate a second dimension, instead of the index of input data, which keeps local distances between input data points, by using the prototypes values (300 x 1) as a mapping weight for each row of reservoir units (1 x 300). We multiply each reservoir's row (1 x300) by the weights of its closest prototype (300 x 1) to get a new one value.

Experiment 1:

In this experiment, we have used 3D artificial data set consisting of 140 data points and contain 6 clusters as shown in Figure 1 top. We have fed the reservoir (300 units) with this data set. Then we have applied our clustering algorithm IEK to cluster reservoir units (140 x 300) into groups. Finally we have plotted the results in 2D as shown in Figure 1 bottom. In Figure 1 bottom left, the first axis represents the index of the data point and the other axis represents the number of the winning prototype. The winning prototype is the closest prototype to the input data point. As shown in Figure1 bottom left, we can see that all groups of data in the input dataset have been projected through reservoir successfully into 2D space. However, the clustering algorithm does not preserve topology between clusters. Investigating this limitation is one of our targets in future work. We can provide a type of keeping local distances within clusters in the low space by mapping the reservoir weights 300 values into only one value. We have used the prototype weights for this mapping and used the resulted values instead of the index of the point axis.

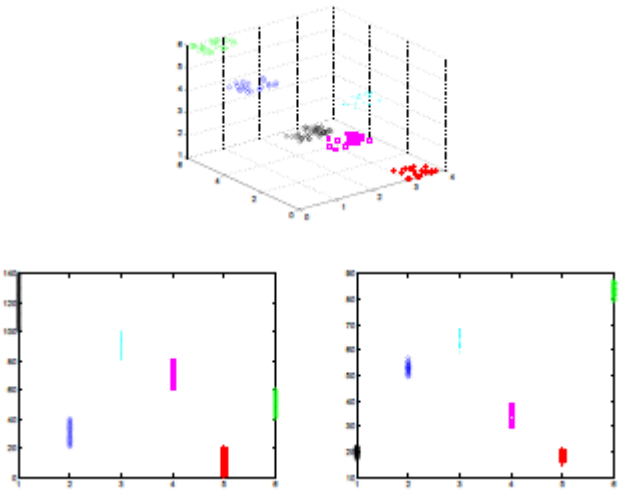

Figure 1. Top: Artificial Data Set consisting of 6 Clusters. Bottom: Projection of the 3D Artificial Data Set using Reservoir-IEK

As shown in Figure 1 bottom right, we can see that the 3D groups of data have been projected successfully into 2D space and local distances within clusters have been presented.

Table 1. The Euclidean Distances between Prototypes in the Reservoir Space. Rows and Columns represent Clusters' Prototypes

	Center of Red *'s	Center of Blue o's	Center of triangles	Center of squares	Center of cyan .'s	Center of black o's
Center of red *'s	0	3.8799	5.6583	1.9600	4.1400	1.9112
Center of blue o's		0	1.9684	2.0463	1.6247	3.1829
Center of triangles			0	3.1829	1.8800	5.1427
Center of squares				0	2.2937	2.1675
Center of cyan .'s					0	4.1575
Center of black o's						0

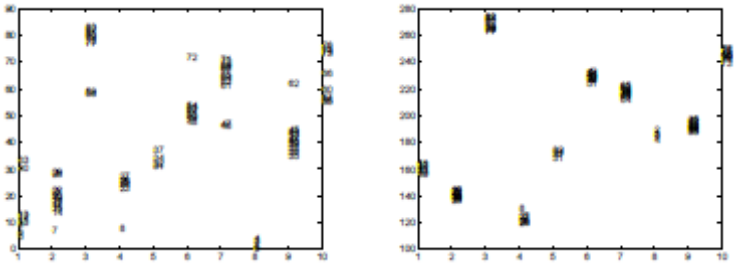

Figure 2. Projection of IE data (1000 samples - January) using the Reservoir-IEK

Also, we can notice a part of preserving the topology when we look to this figure horizon- tally. For example, the black o's and red *'s are closest to each other in the output space. From Table 1, which shows the Euclidean distances between the prototypes in the reservoir space, we can see that they have the same relationship in the reservoir space. Also, in Table 1 we can see that the prototype of green triangles is the farthest from red *'s (or black o's) and this has been reflected in the output space as shown in Figure 1 bottom right.

Experiment 2:

The book "Irrational Exuberance" [20] uses a stock market dataset which represents the closing price each month of the S & P Composite to illustrate its thesis and the author has made this dataset available to all. We have used this closing price, the dividend, earnings, consumer price index and the long term interest rate to form a 5 dimensional data set.

In this experiment, we have fed the reservoir (300 units) with the first 1000 data points of the temporal financial data set IE. Then we have clustered the reservoir units (1000 x 300) using IEK into several groups (10 in this example). Finally we have drawn the results in Figure 2. The numbers in the figure represent the years from 1871 so that for example 1901 is year 30 on the diagram. In Figure 2 left, we used for dimensions the number of the prototype and the index of the input data (or reservoir), while in Figure 2 right, we replaced the index of the input data by the value generated after mapping 300 units to one value using prototype's weights (300).

As shown in Figure 2 left, we can see an interesting projection; consecutive years have been projected close to each other. From Figure 2 left we can notice the following:

• At column 2 (prototype number 2), years 14-22 are grouped together. Also we see years 7, 28, and 29 joined this group. At this stage we do not know if the years 7, 28 and 29 are noise or have

the same characteristics of the years 14-22. In future work, we need to use different methods for visualizing this data set, and investigate and compare the results.

- At column 3, years 77-83 are grouped together. Also years 58 and 59 joined this group.
- At column 4, years 23-27 are grouped together. Also year 8 joined this group.
- At column 6, years 48-54 are grouped together. Also year 72.
- At column 7, years 61-71 except 62 and 66 are grouped together.

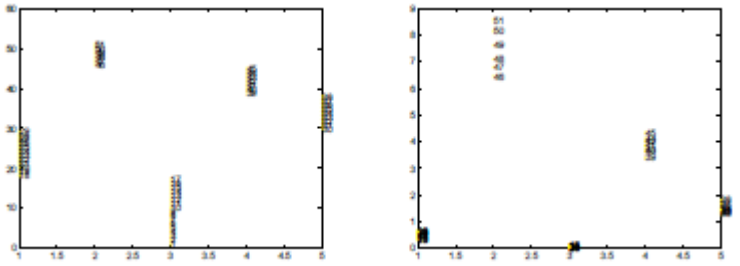

Figure 3. Projection of Airline1 Data Set using the Reservoir-IEK

- At column 8, years 0-4 are grouped together.
- At column 9, years 35-45 except 37 are grouped together.

From Figure 2 right, which shows a part of preserving topology, we can notice the following:

- Years at column 4, 23-27, are farthest from the years at column 3, 77-83.
- Years at columns 1, 2 and 4 are closest to each other, thus we can combine them into a bigger group that includes the consequent years 5-30.
- Years at columns 5 and 9 can be combined together into one group of years 31-45, years at column 8 (0-4) are either noises or they having the same characteristics of the years 31-45.
- Years at columns 6, 7 and 10 can be combined into one bigger group that contains the consequent years 46-76 except 58 and 59.

Experiment 3:

In this experiment, we use another real data set, airline1 dataset. This data set consists of 51 samples, representing the years from 1949 to 1999 (numbers 1-51 on diagram). Each sample has 12 dimensions representing the number of passengers every month January- December. Figure 3 shows the projection of this dataset using reservoir-IEK clustering.

We can see an interesting projection of airline data set in Figures 3. As shown in Figure 3, we have 5 groups, the points in first group (1-17) are closer to each other's than the points in the second group (18-29). The points in the second group are closer to each other than the points in the third group (30-38). Also, if we notice the distances between groups, we can see that the first group is closer to the second group than the second group to the third, in other words, the distance between the first two groups is less than the distance between the last two.

3. Topology Preserving Mapping

In this section, we show how to improve the results in the previous section by providing a type of topology preserving mapping in the projection. We use IEK with the Generative Topographic Mapping (GTM) [7-9] to preserve the topology.

GTM was developed by Bishop as a probabilistic version of the SOM, in order to overcome some of the problems of this map, especially the lack of an objective function. It is a mixture of experts' model which treats the data as having been generated by a set of latent points. The GTM can be described as a non-linear latent variable model that defines a mapping from the latent space to the data space, generating a probability density within the latter.

We have recently used this idea of a latent space [2-4] but with an objective function which is not a probabilistic function and thus is not optimized using the Expectation- Maximization algorithm (EM). Instead, we have developed the Inverse Exponential K- means algorithm (IEK) as the learning process. IEK is more robust to the initial parameters than K-means and the EM algorithm. It is also provides better results regarding convergence to a local optimum.

By adding a latent space model to the IEK algorithm we have created the Inverse- Exponential K-means Topology-preserving Mapping (IEKToM) which we illustrate on the financial data used above.

The basis of our model is K latent points, t1, t2, • • • , t_K , which are going to generate the K prototypes, $\mathbf{m}_k$. To allow local and non-linear modeling, we map those latent points through a set of M basis functions, $f_1()$, $f_2()$, • • • , $f_M()$. This gives us a matrix Φ where $\varphi_{kj} = f_j(t_k)$. Thus each row of Φ is the response of the basis functions to one latent point, or alternatively we may state that each column of Φ is the response of one of the basis functions to the set of latent points. One of the functions, fj (), acts as a bias term and is set to one for every input. Typically the others are gaussians centered in the latent space. The output of these functions are then mapped by a set of weights, W, into data space. W is M × D, where D is the dimensionality of the data space, and is the sole parameter which we change during training. We will use $\mathbf{w}_i$ to represent the i^{th} column of W and Φ_j to represent the row vector of the mapping of the j^{th} latent point. Thus each basis point is mapped to a point in data space, $\mathbf{m}_j = (\Phi_j W)^T$.

We may update W either in batch mode or with online learning. In IEKToM we used the Inverse Exponential K-means algorithm to create a new topology preserving algorithm.

Each data point is visualized as residing at the prototype on the map which would win the competition for that data point. However we can do rather better by defining the responsibility that the j^{th} prototype has for the i^{th} data point as

$$r_{ji} = \frac{\exp(-\gamma \parallel \mathbf{x_i} - \mathbf{w_j} \parallel^2)}{\sum_k \exp(-\gamma \parallel \mathbf{x_i} - \mathbf{w_k} \parallel^2)} \qquad (11)$$

We then project points taking into account these responsiblities: let y_{ij} be the projection of the i^{th} data point onto the j^{th} dimension of the latent space; then

$$y_{ij} = \sum_k t_{kj} r_{ki} \qquad (12)$$

where t_{kj} is the j^{th} coordinate of the k^{th} latent point. When we use these algorithms for visualization purposes, it is these y-values (which are typically two dimensional coordinates) which we use. Note that this method represents each data point $\mathbf{x}_i$ by a value $\mathbf{y}_i$ where $\mathbf{y}_i$ is a weighted sum of the coordinates of the original latent points. An alternative (which is typically used the SOM) is to find the latent point with greatest responsibility for the data point and allocate its $\mathbf{y}_i$ value at this latent point.

3.1. Simulation

Experiment 1:

In this experiment we have fed the reservoir (300 units) with the IE temporal dataset.

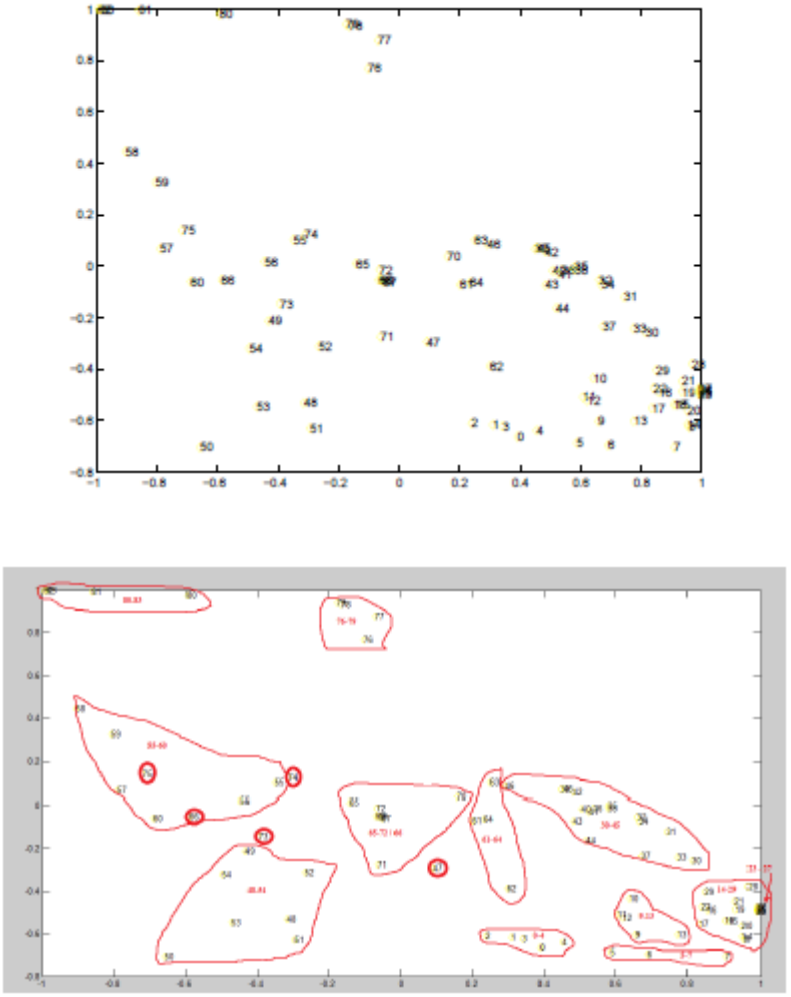

Figure 4. Projection of IE Data (1000 samples - January) using the Reservoir-IEKToM

Then we have projected the reservoir (1000 x 300) onto two-dimensional latent space using IEKToM. Figure 4 shows the result of the projection. In Figure 4 bottom, we can see an interesting projection; consecutive years have been projected close to each other, *e.g.*, 0-4, 14-29 and 30-45. From Figure 4 bottom, we can notice the following:

- Information about this dataset shown in Figure 2 right has been reflected with this projection using IEKToM, *e.g.*, years 23-27 are the farthest from year 76-83.
- Figure 4 bottom shows more information about data than that in 2 right. It shows local distances between data points.
- Some groups of years can be combined into bigger group, *e.g.*, we can combine years 0-45 in one big group.
- It is possible to divide the whole years into 3 big groups 0-45, 46-75 and 76-83.
- Years 55-60 are the closest to years 76-83.

• Years 48-54 at column 6 in Figure 2 right, are close to years 60s and 70s at columns 7 and 10, and far from years 23-27 at column 4. This has been shown with more information in Figure 4 bottom. In this figure we see the relative distances between data points.

Experiment 2:

In this experiment, we use another version of airline dataset, airline2. In this dataset we have 612 data samples with one dimension representing the number of passengers from January 1949 to December 1999. For this data set we make a reservoir 50 units for each data sample to keep the history of time in this temporal dataset. Then we have projected the reservoir units (612 x 50) into 2 dimensional latent space using IEKToM as shown in Figures 5, 6 and 7. To clarify the results, we have drawn the results for each month in a separated figure, after training the whole dataset in the same time.

We have shown in the figures the projections of every month (reservoir space - 50 units) among 51 years (1949-1999). We have an interesting visualization and we can get lots of information about this airline dataset. For example we can notice the following: We have shown in the figures the projections of every month (reservoir space - 50 units) among 51 years (1949-1999). We have an interesting visualization and we can get lots of information about this airline dataset. For example we can notice the following:

- There is a progress in all months and the number of passengers is increasing continuously every year.
- The increasing number of passengers in early years is relatively small comparable to the latter years. For example, if we look to Figure 5 second row, we can notice that the distance between year number 0 and number 25 is shorter than the distance between year number 25 and 50. This means that the rate of increasing number of passengers currently is higher than that rate in the past.
- The point that has the highest value, exists in Figure 6 bottom row, it represents August in 1999 (50). This means that the highest number of passengers recorded is in August, 1999. Also, we can see that July month is always closer to August month in all years. So from the figures we can rate August as the first month regarding passengers' numbers and July as the second. In the other side, from Figure 5 top and second row, we can see that January and February have the smallest values and thus they have the minimum numbers of passengers every year. From Figure 7 bottom, we can also see the number of passengers in December is lower than those in the other months (except January and February).
- Most years have the same structure regarding number of passengers. For example, for every year we can notice that August and July hold the maximum number of passengers among other months while January and February hold the minimum number of passengers. We have shown this structure in Figure 8 by showing the projection of every month in 1999.
- If we combine all figures together, we get Figure 9

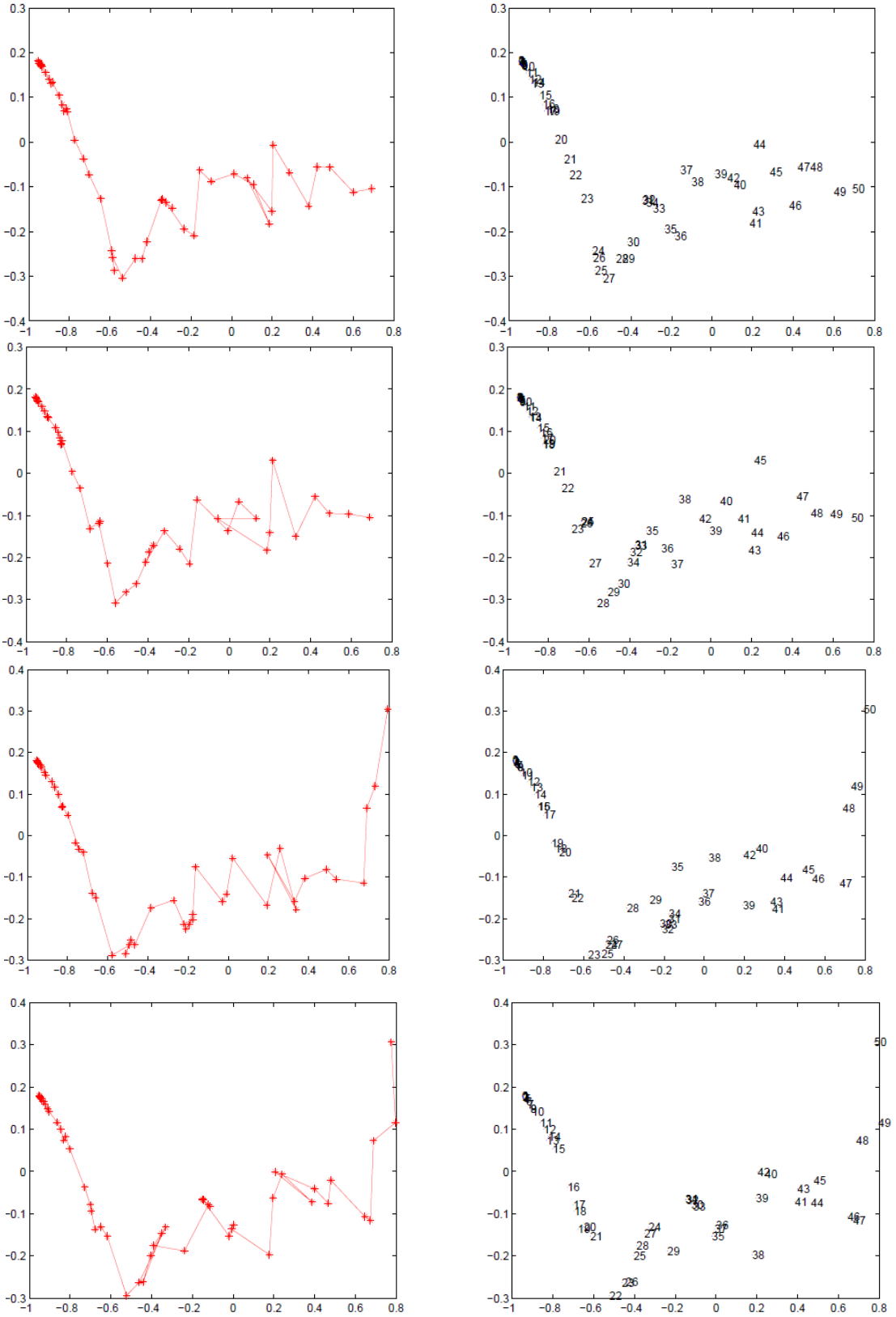

Figure 5. Projection of Airline2 Data using the Reservoir-IEKToM. Top Figures to Bottom Projections of January to April respectively

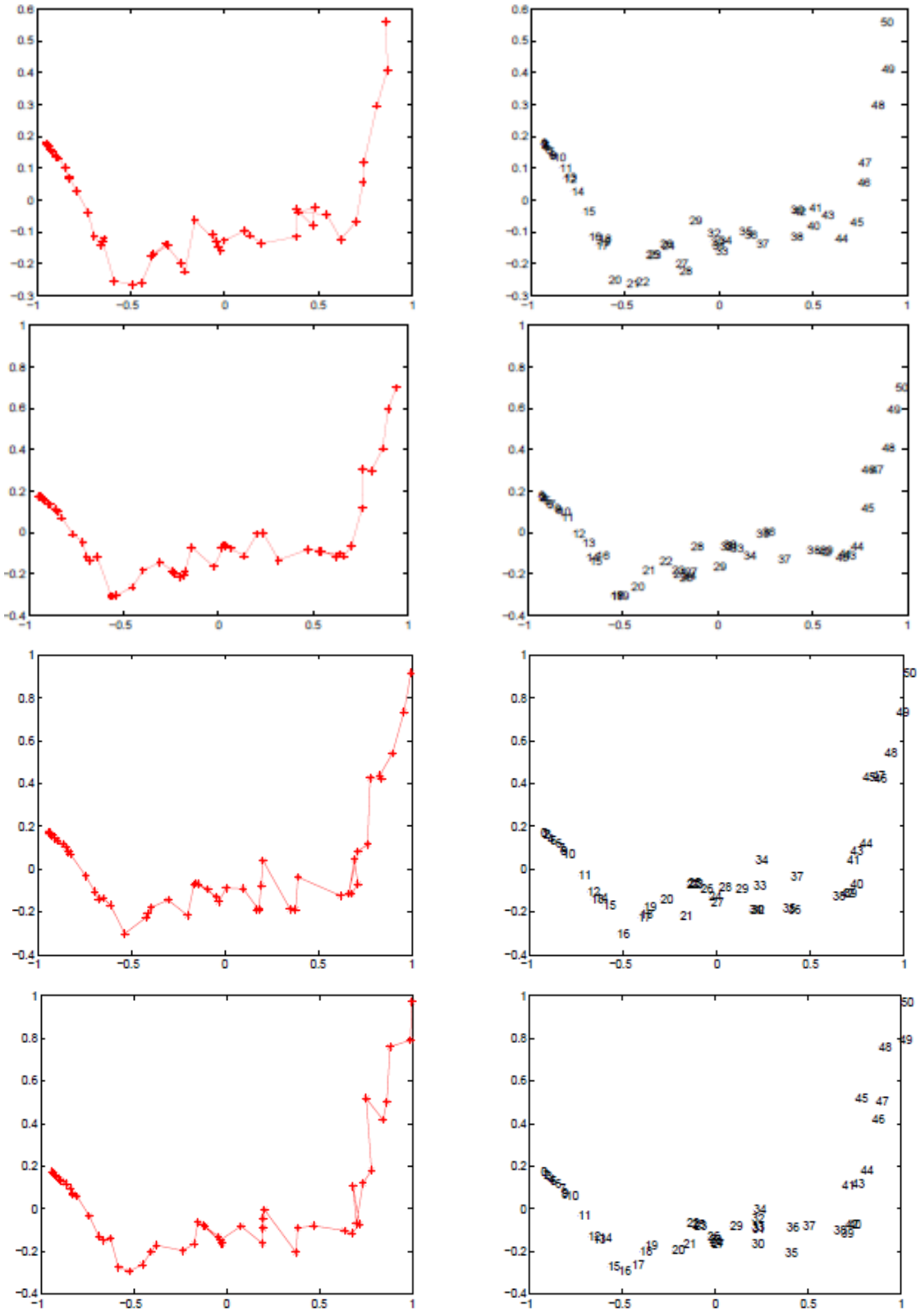

Figure 6. Projection of Airline2 Data using the Reservoir-IEKToM. Top Figures to Bottom Projections of May to August respectively

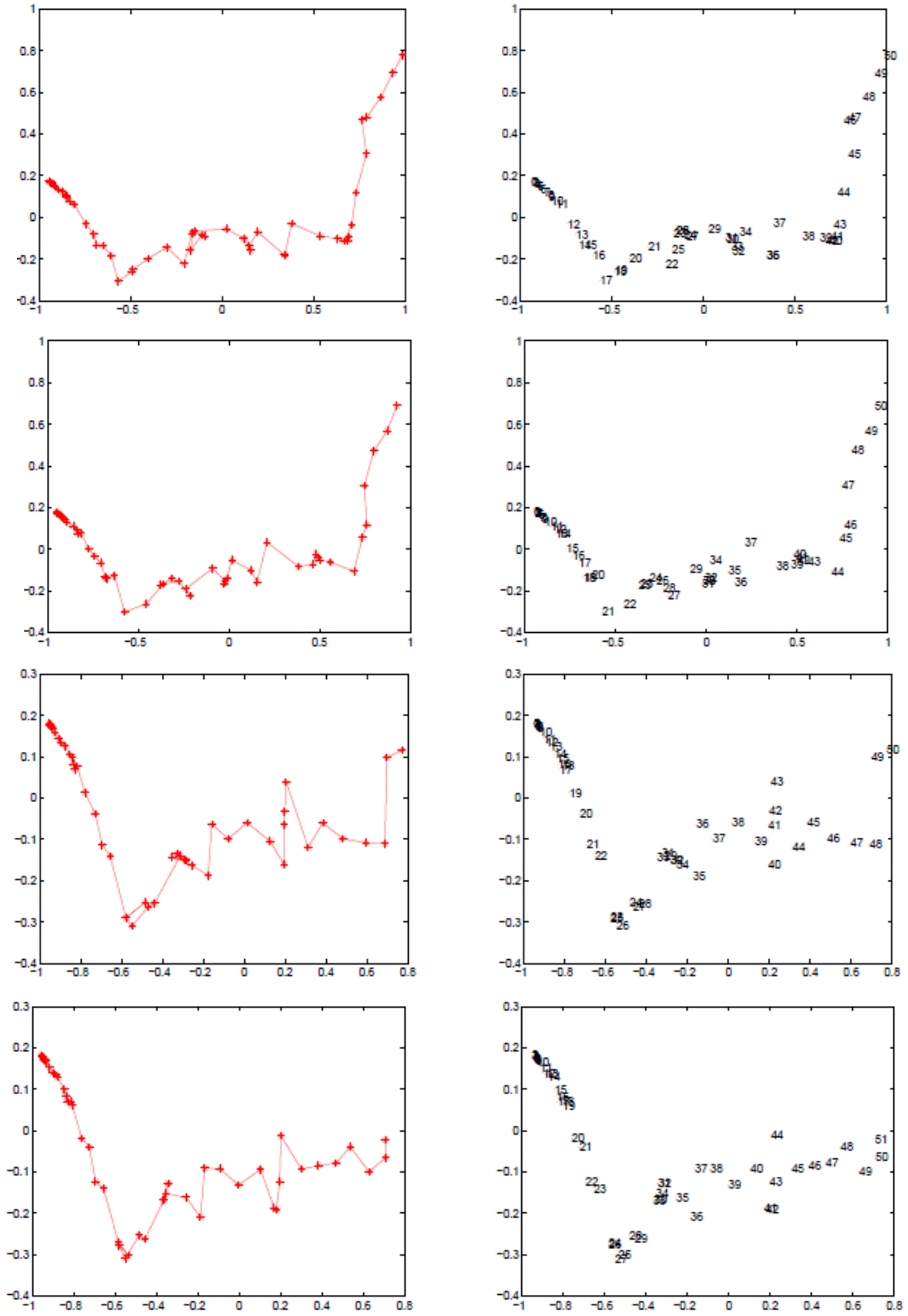

Figure 7. Projection of Airline2 Data using the Reservoir-IEKToM. Top Figures to Bottom are Projections of September to December respectively

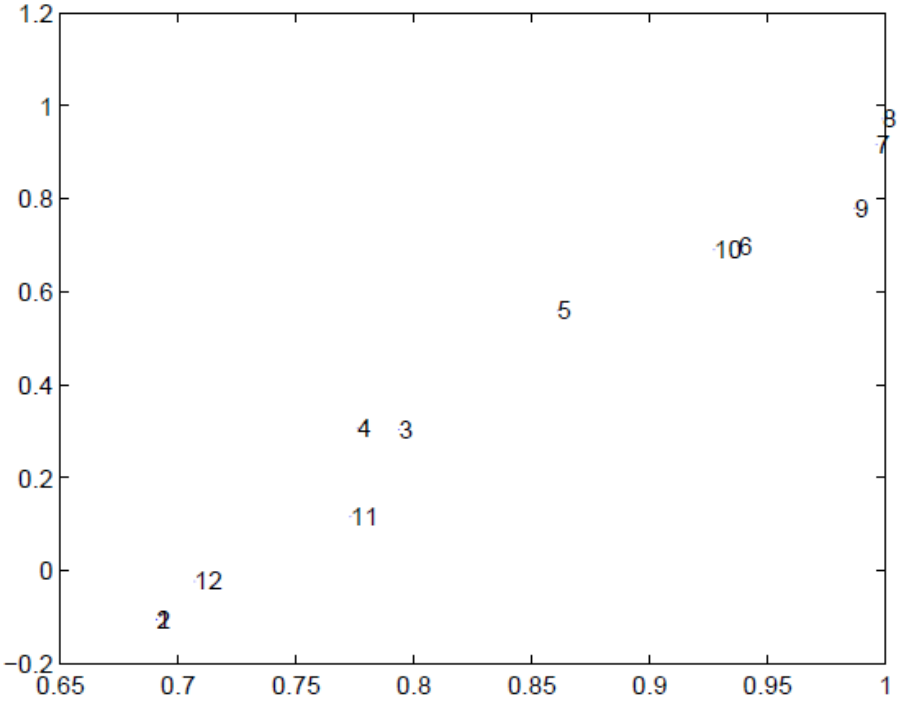

Figure 8. Months only in last year, Projection of Airline2 Data using the Reservoir-IEKToM

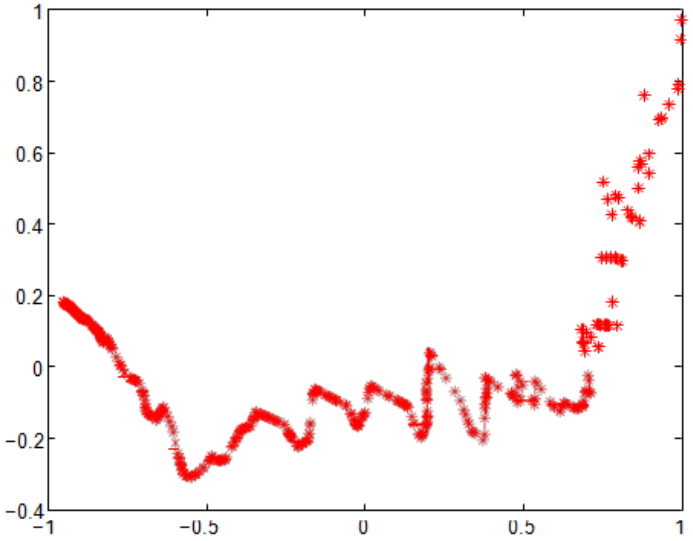

Figure 9. All months together (51 years), Projection of Airline2 Data using the Reservoir-IEKToM

4. Conclusion and Future Work

In this paper we have shown how the clustering algorithm can be used with reservoir to project and visualize temporal dataset. The clustering algorithms provide an advantage when we have groups of data with different and none linear shapes. The clustering can help in separating the non-linear interfering shapes from each other, *e.g.*, BIRCH which depends on density in clustering data. Another advantage of using clustering algorithm is that its speed comparable to other visualizing algorithms. However, the clustering algorithms have limitation in preserving the topology between

data points. Thus we have used IEK clustering algorithm with GTM structure to preserve the topology. Although IEKToM projection has an advantage of preserving the topology, it has limitation in consuming time in processing. Thus in future work we need to investigate this limitation and gain the speed of clustering in topology preserving mapping. We need to investigate and improve our clustering algorithms to be used for topology preserving mapping. Also, we need to test real dataset that contains non-linear shapes and show the advantages of using clustering with reservoir to project and visualize time series data.

Acknowledgements

I would like to extend my sincerest thanks and appreciation to the Arab Fund for Economic and Social Development for their grant and support to accomplish and complete this research.

References

[1] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding", In The eighteenth annual ACM-SIAM symposium on Discrete algorithms, **(2007)**, pp. 1027-1035.
[2] W. Barbakh, M. Crowe and C. Fyfe, "A family of novel clustering algorithms", In 7th international conference on intelligent data engineering and auto- mated learning, IDEAL2006, Springer, ISSN 0302-9743, **(2006)**, pp. 283-290.
[3] W. Barbakh and C. Fyfe, "Local vs global interactions in clustering algorithms: Advances over k-means. International Journal of Knowledge-based and Intelligent Engineering Systems, ISSN 1327-2314, vol. 12, no. 2, **(2008)**, pp. 83-99.
[4] W. Barbakh, Y. Wu and C. Fyfe, "Non-standard exploratory data analysis", Springer, **(2009)**.
[5] M. Celebi, "Effective Initialization of K-means for Color Quantization", In Proc. of the IEEE International Conference on Image Processing, DOI: 10.1.1.151.5281, **(2009)**, pp. 1649-1652.
[6] S. Basterrech, C. Fyfe and G. Rubino, "Initializing echo state networks with topographic maps", In 2nd International Conference on Morphological Computation (ICMC 2011), **(2011)**.
[7] C. M. Bishop, M. Svens´en and C. K. I. Williams. "Gtm: The generative topographic mapping", Neural Computation, vol. 10, no. 1, **(1998)**, pp. 215-234.
[8] C. M. Bishop, M. Svens´en and C. K. I. Williams, "Developments of the generative topographic mapping", Neurocomputing, vol. 21, no. 1, **(1998)**, pp. 203-224.
[9] C. M. Bishop, M. Svensen and C. K. I. Williams, "Gtm: A principle alternative to the self-organizing map", Advances in neural information processing systems, vol. 5, **(1997)**, pp. 354–360.
[10] C. Fyfe, "A scale invariant feature map", Network: Computation in Neural Systems, vol. 7, **(1996)**, pp. 269–275.
[11] C. Fyfe, "Hebbian Learning and Negative Feedback Artificial Neural Networks", Springer, **(2004)**.
[12] C. Fyfe, "Two topographic maps for data visualization", Data Mining and Knowledge Discovery, vol. 14, **(2007)**, pp. 207–224, ISSN 1384-5810.
[13] H. Jaeger, "The echo state approach to analysing and training recurrent neural networks", Technical Report 148, German National Research Center for Information Technology, **(2001)**.
[14] T. Kohonen, "Self-Organising Maps", Springer, **(1995)**.
[15] T. Kohonen, "Self-Organizing Maps", Springer Series in Information Sciences, third edition, vol. 30, **(2001)**.
[16] A. Likas, N. Vlassis and J. J. Verbeek, "The global k-means clustering algorithm", Pattern Recognition, vol. 36, **(2003)**, pp. 451–461.
[17] M. Lukoˇseviˇcius, "On self-organizing reservoirs and their hierarchies", Technical Report 25, Jacobs University, Bremen, **(2010)**.
[18] M. Lukoˇseviˇcius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training", Computer Science Review, vol. 3, **(2009)**, pp. 127–149.
[19] A. Rodan and P. Tiˇno, "Minimum complexity echo state network", IEEE Transactions on Neural Networks, vol. 22, **(2011)**, pp. 131–44.
[20] R. J. Shiller, "Irrational Exuberance", Princeton University Press, **(2005)**.
[21] J. Sun, "Extending metric multidimensional scaling with Bregman divergences", PhD thesis, School of Computing, University of the West of Scotland, **(2011)**.
[22] J. Sun, M. Crowe and C. Fyfe, "Extending metric multidimensional scaling with bregman divergences", Pattern Recognition, no. 44, **(2011)**, pp. 1137–1154.
[23] M. E. Tipping, "Topographic mappings and feed-forward neural networks", PhD thesis, The University of Aston in Birmingham, **(1996)**.
[24] T. D. Wang and X. Wu and C. Fyfe, "Comparative study of visualisation methods for temporal data", 2012 IEEE Congress on Evolutionary Computation, **(2012)**.
[25] T. D. Wang and C. Fyfe, "The role of structure size and sparsity in echo state networks for visualization", The United Kingdom Conference on Computational Intelligence, **(2011)**.
[26] M. Al- Zoubi, A. Hudaib, A. Huneiti and B. Hammo, "New Efficient Strategy to Accelerate k-Means Clustering Algorithm", American Journal of Applied Science, vol. 5, no. 9, **(2008)**, p. 1247–1250.

