Fast and Accurate Neural Multidimensional Scaling and its Application to Map Reconstruction

İ. Kuban Altınel, Ind. Eng. Dept., Boğaziçi Univ., İstanbul, 80815 Türkiye

Necati Aras, Faculty of Management, McGill University, Montreal, Canada

B. John Oommen, Sch. of Comp. Sci., Carleton University, Ottawa, Canada

Abstract

We consider the fundamental problem of reconstructing a map (referred to as *MapRecon*) when the given data is the set of road travel distances among cities in a region. This problem is the "inverse" of the distance estimation problem, in which the goal is to determine a good estimator for inter-city road travel distances as a function of their city coordinates. More specifically, given the road distances among cities in a geographical area, we attempt to determine the *locations* of the cities in a two dimensional map so that the Euclidean inter-city distances approximate the (actual) road distances as closely as possible. The reported solutions to this problem are few, and primarily involve multidimensional scaling (MDS) techniques. We propose to overcome their distinct disadvantages by solving *MapRecon* with a self-organizing method. Our solution, *NeuroMapRecon,* is accurate and does not involve any intricate matrix computations or generalized inverse computations. Furthermore, it is adaptive and can be said to be of a "real time" flavour. The proposed method has been quite rigorously tested on different data sets obtained from various countries. Our results have also been compared with the performance of the Classical MDS and ALSCAL. The accuracy of the proposed method is superior. It has also the following two desirable properties. First, we can obtain point configurations even if some of the input data are missing. Second, it becomes possible to determine configurations where points representing cities are located very close to their original locations.

1 Introduction

One of the classic problems studied in operations research and geographic information systems is the *distance estimation problem*, which involves the determination of a good estimator for the inter-city road distances evaluated as a function of the city coordinates in the map. This problem has been extensively studied because of its importance in location, distribution and logistic problems [1, 2, 3, 4]. The distance estimation problem has an "inverse" problem which we call the *map reconstruction problem* (referred to as *MapRecon* in the following). It is based on resolving the following question: Given the road distances among a set of cities, determine locations for cities where distance relationships are preserved. Observe that this is a rather general problem, since the computed configuration for the cities can be regarded as an abstract map reflecting a hidden structure.

From the above perspective, the reader will observe that *MapRecon* has a close connection with multidimensional scaling (MDS). The main goal of MDS is to visualize and analyze similarities or dissimilarities between "objects" using a lower dimensional representation of their feature vectors, while, in turn, simultaneously preserving the hidden structure.

The first MDS method was proposed by Torgerson [5], and historically has its roots in psychometrics where it was used to discover people's judgments on the similarity of various objects. The literature on MDS is fairly extensive, and reviewing it here is both impractical and unnecessary. In the context of this paper, the excellent book by Borg and Groenen [6] should suffice to bring the reader up to speed.

In terms of nomenclature, if the MDS methodology is based on measured dissimilarities (at the ratio level of measurement), it is called a *metric* MDS. In this case the aim of the exercise is to find a configuration of points that preserves the scaled dissimilarities between the objects. Typically, this scaling is achieved by a transformation function. If the transformation function is linear, the corresponding MDS is described as a *ratio* MDS. As opposed to this, if the dissimilarities are used directly, it is called an *absolute* MDS scheme. If the MDS is based on proximities obtained at the ordinal level of measurement (e.g., judgements, perceptions), the scheme is referred to as a *nonmetric* MDS. In nonmetric MDS [7] we assume that there is a monotonic relationship between the inter-point distances and observed proximities. This, in turn, means that the reproduced distances (namely, those which are obtained after locating the "objects" in the lower dimensional space), are constrained to preserve the rank order among the similarities or dissimilarities rather than the actual distances among objects. It is important to note that a metric MDS scheme can be used to approximate nonmetric MDS by using the rank orders as dissimilarities [6].

MapRecon is equivalent to the metric MDS where objects are cities in a given region, and dissimilarities among objects are considered to be road distances among the cities. To formally formulate the problem, we first of all observe that if we are given a matrix $\Delta = [\delta_{ij} : i, j = 1, ..., n]$ of road distances between all pairs of n cities, these cities can be represented by n points in a Euclidean space E^{n-1} such that the Euclidean inter-city distances d_{ij} are equal to δ_{ij} for all i, j provided that the elements of Δ satisfy the metric inequality. The problem becomes both pertinent and non-trivial if we have to represent the cities by n points in a lower dimensional subspace E^r where $r << n-1$. It is clear that in the case of *MapRecon,* $r = 2$. An analytical technique, referred to as the classical MDS [8] has been shown to be capable of exactly reproducing the road distances in the case when they are Euclidean (i.e., $\delta_{ij} = d_{ij}$). Indeed, with a little imagination, this technique can also be employed when the road distances are approximately Euclidean. Since typical road distances can be perceived as being Euclidean distances distorted by errors, classical MDS has been used to solve the map reconstruction problem [9]. However, such solutions possess a fundamental "infirmity", namely, that the reproduced inter-city distances deviate *significantly* from the true road distances, resulting in a solution with poor accuracy. In this context, accuracy is measured in terms of the discrepancy between actual road distances and their estimates obtained by MDS.

Apart from the considerations mentioned above, MDS techniques possess a few other important drawbacks. The first one is related to the number of input dissimilarities or distances that are necessary to determine a feasible and meaningful location for the objects. Classical MDS algorithms require an $n \times n$ matrix of input dissimilarities, where n is the number of objects. This could be problematic because of the resource limitations (cost, time etc.) involving the data collection procedure. Another important fact is that a map created by *any* MDS method is usually orientation-free, that is the orientation of the points representing the objects is arbitrary. Generally speaking, this does not pose a significant problem since most of the time we are only concerned with the proximity of the reproduced points on the lower dimensional map, which enables us to "visually" discover the hidden structure in the data. There may be occasions, however, where apart from the dissimilarity that exists between the "objects", we have further information about some of the objects. This information could be physical measurements about the objects along a set of dimensions such as the latitude and longitude of cities. In such a case the objective becomes to discover a relation between the coordinates of the objects (possibly in a high dimensional space) and the positions of the points representing these objects in a lower dimensional space by preserving the proximities. Steyvers [10] gives an example of this in a psychological context. He points out that when the physical representation of the features comprising the stimuli is ignored, as in a traditional MDS technique, it becomes difficult to *interrelate* the actual positions of stimuli with the points on the map that represent the stimuli. This happens because the orientation of the points on the reproduced map turns out to be different from the original map of objects due to the possible translation, rotation and reflection phenomena.

1.1 Contributions of This Work

In this work we propose a new self-organizing technique for solving *MapRecon*, which does not involve any matrix computations. If the method is provided with the geographical locations of a few (at most *two*) cities, referred to as the *anchor* cities, the resultant map has numerous properties. The specification of a single *anchor* city yields a map that is an element of the set of translation invariant maps. Maps that belong to this set, however, are subject to rotation and reflection. The specification of *two anchor* cities results in a map which is an element of the set of translation and rotation invariant maps. On the other hand, each map in this set is subject to reflection. If, however, the specified *pair of anchor* cities lie on the convex hull of the map and the neurons are initially positioned on the same side of the line joining these two anchor cities with the original cities, the resultant map becomes invariant to translation, rotation and reflection. We are not aware of any reported map reconstruction method which guarantees these advantages.

In addition to the above, our solution to *MapRecon*, *NeuroMapRecon*, has the capability of reproducing a map with the above invariance properties, even if some elements in the inter-city distance matrix are missing. In other words, complete distance matrix is not required to find a solution by *NeuroMapRecon*. In this respect, *NeuroMapRecon* is both novel and pioneering—no other reported scheme accomplishes this.

NeuroMapRecon has been implemented and tested rigorously on several data sets, where each set consists of the inter-city road distances in a given geographical region. The results are truly impressive. In particular, we have compared the performance of the new method with that of the classical MDS and ALSCAL[1] [11]. Our experimental results clearly demonstrate that the performance of the new method is better with respect to the scaling error measured by the stress function. Furthermore, the final locations of the cities tend to overlap with the original locations when the previously mentioned simple strategies are incorporated with the learning updates.

It is also pertinent to point out that the method we use adaptively migrates the locations of the cities on the plane using techniques akin to those used in Vector Quantization (VQ) [12, 13] and Kohonen's Self Organizing Map (SOM) [14, 15]. However, the VQ and SOM methods reported in the literature are not directly applicable because, unlike in the latter, concepts such as "winning neuron", "bubble of activity", or "neighborhood" do not exist. Indeed, as we shall presently clarify, while the fundamental migration concepts are utilized, the issues that relate VQ and SOM with the latter concepts, have to be, unfortunately, discarded. It is amazing that we can obtain convergence even though these fundamental concepts are made irrelevant.

1.2 Implications of These Results in Pattern Recognition

The reader will easily observe that these present results have an overall significance in the general fields of pattern recognition, contour and image processing, and geographical information systems. However, viewed from the perspective of the more recent contributions of Duin and Pekalska [16, 17, 18], the present paper has also implications in designing *zero-error* pat-

[1] In the experiments we used ALSCAL that is available in SPSS 10.0 for Windows.

tern recognition systems that utilize dissimilarity representations[2]. More specifically, Duin and Pekalska [16, 17, 18] have recently proposed brilliant concepts which can hopefully open the doors for "zero-error" pattern recognition. Although the ideas motivating the new directions are fairly complex, they can be crystallized as follows. Rather than specify the various samples (the training and testing objects from the various classes) in terms of their "features", dissimilarity-based pattern recognition processes only the dissimilarities between the objects. At the outset, the question of how these dissimilarity indices are computed is irrelevant, and is therefore not considered. Given *only* the set of dissimilarities between the training samples, this new field of pattern recognition attempts to locate the samples in a "hypothetical" space where the points are located in such a way that their relative dissimilarities is reflected by the proximity of the points that they represent. Pekalska and Duin resorted to MDS methods to achieve such a representation. We contend that our new results have direct applications in this domain.

In the next section, we briefly describe classical MDS and ALSCAL, and highlight their properties. The *NeuroMapRecon* is introduced in Section 3. Section 4 includes experimental results on the performance of the new method. Finally, Section 5 concludes the paper.

2 Map Reconstruction With Classical MDS and ALSCAL

If we know the coordinates of two points in a multidimensional coordinate system, the Euclidean distance between these two points is easily calculated. If the p-dimensional coordinates of n points are given in an $n \times p$ dimensional matrix, say $\mathbf{X}$, where each column corresponds to a dimension, and each row corresponds to a point, then the Euclidean distances between all pairs of points can be calculated by using the entries of $n \times n$ dimensional matrix $\mathbf{B} = \mathbf{X}\mathbf{X}^T$ where $b_{rs} = \sum_{j=1}^{p} x_{rj} x_{sj}$. The squared Euclidean distance between any two points r and s is:

[2]The third author is very grateful to Professor Bob Duin for his fascinating talk at the PRIS 2002 Workshop in Alicante, Spain, in April 2002, and the instructive discussions they had at that time.

$$
\begin{aligned}
d_{rs}^2 &= \sum_{j=1}^{p} \left(x_{rj} - x_{sj}\right)^2 \\
&= \sum_j x_{rj}^2 + \sum_j x_{sj}^2 - 2\sum_j x_{rj}x_{sj} \\
&= b_{rr} + b_{ss} - 2b_{rs}
\end{aligned} \tag{1}
$$

When we put the center of gravity of the points at the origin, $\sum_{r=1}^{n} x_{rj} = 0$, $j = 1, ..., p$, the sum of any row or column of $\mathbf{B}$ will be zero. After some algebraic manipulations it may be shown that

$$
b_{rs} = -\frac{1}{2}\left(d_{rs}^2 - d_{\cdot s}^2 - d_{r\cdot}^2 + d_{\cdot\cdot}^2\right) \tag{2}
$$

where $d_{\cdot s}^2$ and $d_{r\cdot}^2$ denote the average of the elements in column s and row r respectively, and $d_{\cdot\cdot}^2$ is the average of all the elements. Since $\mathbf{B}$ is symmetric, we can write $\mathbf{B} = \mathbf{Q}\Lambda\mathbf{Q}^{-1}$ or $\Lambda = \mathbf{Q}^{-1}\mathbf{B}\mathbf{Q}$ where $\mathbf{Q}$ is the matrix whose columns are the eigenvectors of $\mathbf{B}$. If $\mathbf{B}$ is positive definite, all the eigenvalues are greater than zero and it follows that

$$
\mathbf{B} = \mathbf{Q}\Lambda\mathbf{Q}^{-1} = \mathbf{Q}\Lambda^{1/2}\Lambda^{1/2}\mathbf{Q}^{-1} = \mathbf{X}\mathbf{X}^T \tag{3}
$$

Hence the coordinates of n points may be found by setting $\mathbf{X} = \mathbf{Q}\Lambda^{1/2}$. The steps below summarize the procedure:

1. Form the matrix $\mathbf{E}$ with $E_{ij} = -\frac{1}{2}\delta_{ij}^2$ where δ_{ij} is the measured dissimilarity (or distance) between points i and j.

2. Subtract from each element of $\mathbf{E}$ the means of the row and column in which it is located, and add to it the mean of all elements of $\mathbf{E}$. Denote the resulting matrix as $\mathbf{B}$.

3. Find the eigenvalues λ_j and normalized eigenvectors $\mathbf{v}_j$ of $\mathbf{B}$. The coordinates of the n points on the j^{th} axis is given by $\sqrt{\lambda_j}\mathbf{v}_j$. Since we are interested in representing the points in two-dimensional Euclidean space, we choose the largest two eigenvalues.

If the distances obey the triangular inequality, $\mathbf{B}$ will be positive semidefinite. As mentioned before, unless the road distances are exactly Euclidean, there will always be a discrepancy between the reproduced (Euclidean) distances and true road distances. In most cases, road distances among cities in a region are larger than the Euclidean distances because of natural barriers such as lakes, rivers, mountains etc. In order to assess the performance of any other MDS technique we have to use a goodness-of-fit criterion that measures the deviation between the Euclidean distances among points on the map and original distances among objects. One of the most widely used measures is the stress function. There are different stress functions in the literature, normalized stress, raw stress, Kruskal stress and S-stress. We will use the Kruskal stress (also known as "Stress formula 1") which is provided in the output of ALSCAL. It was first proposed by Kruskal [19] in 1964 when he coined the term "stress" (*st*andardized *re*sidual *s*um of *s*quares). The formula for the stress is as follows:

$$\sqrt{\frac{\sum_{i<j}\left(f\left(\delta_{ij}\right)-d_{ij}\right)^2}{\sum_{i<j}d_{ij}^2}} \tag{4}$$

In this formula d_{ij} refers to the Euclidean distance between points i and j on the map, $\|\mathbf{x}_i-\mathbf{x}_j\|$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the coordinates of points i and j. $f\left(\delta_{ij}\right)$ is a linear transformation function. Note that when the original distances are preserved perfectly for all city pairs i and j, then $f\left(\delta_{ij}\right)=d_{ij}$ for all i,j and the stress is zero. Hence, a smaller stress value indicates a better representation. In the case of classical MDS, the function f is the identity function, i.e., $f\left(\delta_{ij}\right)=\delta_{ij}$. In such a case the stress function becomes

$$\sqrt{\frac{\sum_{i<j}\left(\delta_{ij}-d_{ij}\right)^2}{\sum_{i<j}d_{ij}^2}} \tag{5}$$

In other words, they are absolute MDS techniques. ALSCAL, on the other hand, uses a linear transformation function $f\left(\delta_{ij}\right)=\alpha\delta_{ij}$ for scaling the original distances. The optimal value of α is found by setting the derivative of the stress function with respect to α equal to zero and solving for α. It gives $\alpha=\left(\sum_{i<j}\delta_{ij}d_{ij}\right)/\left(\sum_{i<j}d_{ij}^2\right)$. When this term is inserted

into the stress function, it becomes

$$\sqrt{1 - \frac{\left(\sum_{i<j} \delta_{ij} d_{ij}\right)^2}{\left(\sum_{i<j} \delta_{ij}^2\right)\left(\sum_{i<j} d_{ij}^2\right)}} \tag{6}$$

In Section 4, stress given with formula (6) will be used as the basis of comparison among the methods.

3 The Neural Solution: NeuroMapRecon

3.1 Motivating Philosophy: VQ and SOM Strategies

Our aim is to determine a two-dimensional configuration for a set of n points in such a way that the inter-point Euclidean distances approximate the inter-city road distances as much as possible. To achieve this, we will not use any matrix-based computations. Rather, we will perform operations on the points that "resemble" the operations done in vector quantization (VQ) and Kohonen's Self-Organizing Maps (SOM).

The foundational ideas motivating VQ and the SOM are the classical concepts that have been applied in the estimation of probability density functions. Traditionally, distributions have been represented either parametrically or non-parametrically. As opposed to the parametric model of computation in which the distributions are modeled as functions with estimated parameters, in non-parametric methods the data are processed in its entirety. The corresponding resulting pattern recognition algorithms are generally of the nearest neighbor (or k-nearest neighbor) philosophy, and are thus computationally expensive.

The concept of VQ [12, 13] can be perceived as one of the earliest compromises between the above two schools of thought. Rather than representing the entire data in a compressed form using only the estimates, VQ opts to represent the data in the actual feature space, by compressing the information using a "small" set of vectors, called the code-book vectors. These code-book vectors are migrated in the feature domain so that they collectively represent the distribution under consideration. In both VQ and the SOM the polarizing algorithm is repeatedly presented with a point i from the set of points of a particular class.

Each point i is represented by its feature vector $\mathbf{u}_i$. The neurons attempt to incorporate the topological information which is present in $\mathbf{u}_i$. This is done as follows. First of all, the closest neuron to i at time t, j^*, is determined (also called the "winner"). It is represented by the vector $\mathbf{y}_{j^*}(t)$. Here t is the discretized (synchronized) time index. This neuron and a group of neurons in its neighborhood, $B_{j^*}(t)$, are now moved in the direction of $\mathbf{u}_i$. The set $B_{j^*}(t)$ is called the "bubble of activity" at time t. The actual migration of the neurons is achieved by rendering the new point $\mathbf{y}_j(t)$ representing neuron j, namely $\mathbf{y}_j(t+1)$, to be a convex combination of $\mathbf{u}_i$ and every $\mathbf{y}_j(t)$, $j \in B_{j^*}(t)$. More specifically, the updating algorithm for $\mathbf{y}_j(t)$ is as follows:

$$\mathbf{y}_j(t+1) = (1-\alpha(t))\mathbf{y}_j(t) + \alpha(t)\mathbf{u}_i \quad \forall\, j \in B_{j^*}(t). \tag{7}$$

This basic algorithm has two fundamental parameters, $\alpha(t)$ and the size of the bubble of activity $B_{j^*}(t)$. $\alpha(t)$ is called the adaptation constant and satisfies $0 < \alpha(t) < 1$. Kohonen and others [14, 15] recommend decrementing $\alpha(t)$ linearly from unity for the initial learning phase and then assigning a small number to $\alpha(t)$, e.g., 0.2 and further decreasing its value linearly for the fine-tuning phase.

The bubble of activity, $B_{j^*}(t)$, is the parameter which makes VQ differ from the SOM. Indeed, if the size of the bubble is always set to be zero, only the closest neuron is migrated, yielding a VQ scheme. However, in the SOM, the nearest neuron and the neurons within the bubble are also migrated, and it is *this* widened migration process which permits the algorithm to be both *topology preserving* and self-organizing. The size of the bubble is initially kept fairly large to allow a global ordering to develop. Subsequently all the neurons tend to tie themselves into a knot for a value of $\alpha(t)$ that is close to unity; they quickly disperse. Once this coarse spatial resolution is achieved, the size of the bubble is steadily decreased. Thus, only those neurons which are the most relevant to the processed input point will be affected by it, and the ordering which has been achieved by the coarse resolution is not disturbed, but the fine tuning on this ordering is permitted.

3.2 Why VQ and SOM Cannot be Directly Applicable to *MapRecon*

Elegant though they are, VQ and SOM methods cannot be directly applied to the *MapRecon* problem. The reasons for this are the following. First of all, unlike in the fields of VQ and SOM, there is no question of presenting a city (a data point) to the network. This is because, there is no available data point. All that is available is the set of inter-city road distances. Secondly, as a consequence of the latter, there is no question of a "winning neuron" which wins the competition. This, further disallows the entire issue of defining "neighbor neurons" and a "bubble of activity". Thus, we are faced with the issue of tackling the problem of migrating the neurons which represent the cities, without having the luxury of using the very concepts which render such a migration expedient. Indeed, it is amazing that we can obtain convergence even though these fundamental concepts are made irrelevant.

3.3 How *NeuroMapRecon* Utilizes Neural Migration

NeuroMapRecon operates by representing each city by a neuron in a two dimensional space. As a preprocessing step, the elements δ_{ij} of the distance matrix are normalized by dividing them by the largest element in the matrix. The initialization of the neurons are performed by assigning them weights (coordinates) that are uniformly distributed on the unit square (i.e., in the interval $[0, 1]$ along the x and y dimensions). The migration occurs as follows.

At each iteration of *NeuroMapRecon*, a random pair of cities (i, j) is selected from among n cities. The road distance between these two cities is compared with the estimate obtained by calculating the Euclidean distance $d_{ij}(t)$ between the weights of neurons i and j representing cities i and j. If the distance between neurons i and j[3] is less than the road distance between cities i and j, i.e., $d_{ij}(t) < \delta_{ij}$, the neurons are clearly "too close for comfort". To compensate for this, the weights of the neurons are updated. To put it differently, the coordinates of the neurons are moved away from each other along the line connecting them by an amount proportional to the difference between $d_{ij}(t)$ and δ_{ij}. The proportionality factor

[3] The distance between two neurons is defined as the Euclidean distance between the weights of these neurons.

is a parameter called the "step size", $\mu(t)$. As a result, both neuron i and j are moved by an amount equal to $\mu(t)(\delta_{ij} - d_{ij}(t))/2$ so that the distance between the neurons increases by $\mu(t)(\delta_{ij} - d_{ij}(t))$. The value of $\mu(t)$ is reduced once all $n(n-1)$ pairs of cities are presented and the corresponding neurons are updated as it is traditionally done in the case of VQ and SOM. One such iteration is referred to as an epoch of *NeuroMapRecon*. In our implementation the initial value of $\mu(t)$ is equal to unity and it is decreased linearly according to the following rule: $\mu(t) = 1 - t/1000$. If the distance between neurons i and j is too far apart, the weights of the neurons are updated by an amount equal to $\mu(t)(d_{ij}(t) - \delta_{ij})/2$ such that the distance between neurons i and j and the corresponding road distance come closer to each other by $\mu(t)(d_{ij}(t) - \delta_{ij})$.

The reader should observe a fundamental difference between this algorithm and the generic family of VQ and SOM algorithms. That is, we do not have a set of cities which can be considered as sample points being presented to the neurons. Note that we do not use the diagonal elements $\delta_{ii} = 0$ in the updates, and that in each cycle every pair is introduced twice to the algorithm in order to increase the rate of convergence. The termination criterion is based on the improvement in the stress of the configuration. If the reduction of the stress value in two consecutive cycles turns out to be less than a predetermined small value ϵ, the algorithm terminates and the current configuration of the neurons is accepted as the solution. The steps of the method are given below formally.

Input: Scaled road distance matrix $\Delta = [\delta_{ij}, 1 \leq i, j \leq n]$ and $\epsilon = 10^{-6}$

Output: Spatial representation of cities in E^2

Step 0 Generate n neurons with $\mathbf{y}_i(0) \in (0,1)$, $1 \leq i \leq n$.

$stress(0) \leftarrow \infty$ and $t \leftarrow 1$

Step 1 **For** each city pair i and j

Do the following in a random order:

If $d_{ij}(t) \leq \delta_{ij}$, move i and j away from each other by an amount $\mu(t)(\delta_{ij} - d_{ij}(t))/2$

If $d_{ij}(t) > \delta_{ij}$ move i and j towards each other by an amount $\mu(t)(d_{ij}(t) - \delta_{ij})/2$

Step 2 **If** $stress(t) \leq stress(t-1)$ and $stress(t-1) - stress(t) < \epsilon$ **Then** stop
Else $t \leftarrow t+1$, decrease $\mu(t)$ and **GoTo** Step 1

3.4 How *NeuroMapRecon* Enforces Neural Migration

The algorithm described above, clearly, has the effect that it moves *every* pair of neurons in such a manner that their Euclidean distance *after the migration* is closer to the road distance than it was before the migration. Thus, if the road distances are themselves Euclidean, as assumed in the classical MDS, it is possible that the neurons will converge to a configuration for which the stress is zero. Numerical results supporting this claim are provided in the next section.

We now consider the problem of determining a mapping from a set of physical measurements obtained for certain characteristics of objects to their positions in the lower dimensional space. In the feature mapping approach described in [20], apart from the dissimilarity data, the method is permitted to utilize additional information about the object which can lead to configurations that precisely pinpoint the actual feature vectors without translation, rotation and reflection in the obtained configuration. In the context of map reconstruction, this means that if the coordinates of some cities are available as additional information, the configuration found by *NeuroMapRecon* can be made translation, rotation, and reflection invariant. For example, when the Cartesian coordinates of *any* city is known, the neuron representing that city can be uniquely specified, with the additional constraint that the weights of this neuron are not allowed to be updated. Thus if $\|\mathbf{y}_i(t) - \mathbf{y}_j(t)\| = d_{ij}(t) < \delta_{ij}$, where i is the neuron fixed at $\mathbf{y}_i(t) = \mathbf{u}_i$, we enforce the rule that we are only permitted to move j away from i by an amount $\mu(t)(\delta_{ij} - d_{ij}(t))$. Fixing neuron i at the location of city i has the effect of adding a new constraint to *MapRecon*, which gives rise to a configuration where translation is avoided. This strategy is called the "one-fixed" strategy (1FS). However, the maps are still subject to rotation and reflection.

When the coordinates of two cities (coordinates $\mathbf{u}_i$ and $\mathbf{u}_j$ of anchor cities i and j) are available as additional information, the coordinates (weights) of the corresponding neurons can be fixed. In other words, neurons i and j are not updated at all. When the road distance between either of these two cities and another city k is compared with the current

Euclidean distance between the neurons representing the cities (i.e., δ_{ik} or δ_{jk} is compared with $d_{ik}(t)$ or $d_{jk}(t)$, respectively), only neuron k undergoes an update. Consequently, fixing the weights of two neurons prevents not only the translation but also the rotation in the final configuration with respect to the original orientation of the cities. This strategy is referred to as the "two-fixed" strategy (2FS). However, reflection phenomenon can still be observed.

Now we are only left with the question whether it is possible to prevent reflection. Indeed, when we have the additional information that the two anchor cities are located on one of the edges of the convex hull of the cities such that the remaining cities are located on exactly one side of the line connecting these anchor cities, *NeuroMapRecon* provides a final configuration that is also reflection invariant and very close to the actual city orientation. The only requirements are that the anchor cities should not be very close (otherwise rotation may occur) and the remaining neurons are initialized on the same side of the line joining the anchor cities where all other cities are located. This strategy is called the "line-fixed" strategy (LFS). Recall that when there is no additional information on city coordinates, *NeuroMapRecon* is invoked with no fixed neurons. Therefore, this case is designated as the "no-fixed" strategy (NFS).

In the next section where numerical results are presented, we use a performance criterion referred to as the *average location error* in order to measure the deviation between the reproduced map and the original map of the cities. It is defined as follows:

$$LE = \frac{\sum_{i \in S} e_i}{n}. \tag{8}$$

Here, e_i is the Euclidean distance between city i and neuron i which represents city i on the reconstructed map, n is the number of cities, and $\mathcal{S}$ is the set of cities where $|\mathcal{S}| = n$. It is important to note that the location error can only be computed if the city coordinates are available as part of the data. In real-life applications where MDS is used, we do not have access to this information. Otherwise MDS would not be necessary. Our motivation in using this error measure is to show the value of the extra information (the coordinates of at most two cities) in reproducing maps as faithfully as possible to actual maps. Notice that when the input distance matrix consists of Euclidean distances *and* we obtain a city configuration with no distortions such as translation, rotation and reflection, the final coordinates of the neurons will overlap with the cities they represent. In other words, the positions of the cities in the reconstructed map will exactly match with the locations of the cities in the actual map. In such a case, $e_i = 0$ for all $i \in \mathcal{S}$ resulting zero location error. However, this is only possible if the inter-city distances are Euclidean. This means that it is not possible to have a zero location error with a distance matrix consisting of road distances. Therefore reconstruction methods with small location errors are capable of producing maps preserving not only distance relations but also locational proximity.

4 Experimental Results

Using six different data sets consisting of inter-city distances from various geographical locations, we have assessed the performance of three methods, i.e., classical MDS, ALSCAL and *NeuroMapRecon*. Each method is employed to reproduce the locations of the cities for each data set. In the interest of nomenclature, the data sets are referred to by the name of the country from which they are sampled [21]. With the exception of Türkiye data, the number of cities in each data set is 15 resulting in a 15×15 distance matrix. The data for Türkiye consists of inter-city distances among 80 cities, which gives rise to an 80×80 distance matrix. We have two kinds of inter-city distances, Euclidean distances and true road distances among the cities.

The performance of the three methods are evaluated based on two criteria. The first one is the stress given with formula (6). It is a very frequently used measure in the MDS literature, and also one of the two measures employed in ALSCAL (the other is S-stress). The other performance measure is the average location error given in equation (8).

It is possible to divide our experiments into three groups. In the first one, the road distances are Euclidean. The second set of experiments is carried out when the road distances are true. In many real life applications some of the inter-city distances could be missing. Considering this, we perform a third group of experiments where the distance matrix is not complete. First 5, then 10 entries of the distance matrix are chosen randomly and they are not used as an input to the methods with the exception of the classical scaling method since the latter is not designed to handle missing input data. Therefore, we compare

NeuroMapRecon only with ALSCAL in this case.

In all of the experiments the learning rate $\mu(t)$ is decremented linearly starting at unity, according to the update equation $\mu(t) = 1 - t/1000$. Clearly this allows for 1000 or less epochs.

4.1 Results when Road Distances are Euclidean

In each of the cases, *NeuroMapRecon* was employed to reconstruct the map when the road distances among the cities are considered to be Euclidean. The stress values reported in Table 1 demonstrate the accuracy of the new method. Recall that, as the distances are Euclidean, classical MDS and ALSCAL guarantee final configurations with zero stress. Therefore, they are not given in Table 1. Since the final configuration obtained by *NeuroMapRecon* depends on the initial weights (coordinates) of the neurons, we performed 10 replications for NFS, where neurons were assigned random initial weights in each case. The numbers of the first row of the table are the average stress values. Note that this is the case when no extra information about the final configuration is available. For 1FS we randomly determine 10 different cities. In each run we fix the neuron that represents the selected city. Similar to the case with NFS, depending on the initial location of the neurons different final configurations may be obtained even if the same neuron is fixed. Therefore, for each run we performed 10 replications with different initial weights assigned to the neurons. Note that the weight of the fixed neuron remains the same over the replications. As a result, we report in the table the average of 100 replications for 1FS. Likewise, the row associated with 2FS contains average stress values of 100 replications. In each run, first a pair of neurons representing the anchor cities is fixed, and then 10 replications are carried out with different initialization for neuron weights. For LFS again two anchor cities are selected with the extra condition that they are located on one of the longer edges of the convex hull of the cities. The neurons corresponding to the anchor cities are fixed and 10 replications are made by randomly initializing the coordinates of the neurons other than the fixed ones. Thus, the stress values associated with line-fixed strategy are averages of 10 runs. The results obtained by *NeuroMapRecon* are very close to zero showing the success of the method. It undoubtedly performs as well as classical MDS and ALSCAL on the Euclidean data.

	Australia	Canada	England	France	Türkiye	USA
NFS	$4{\times}10^{-8}$	0.0024	$4{\times}10^{-8}$	$4{\times}10^{-8}$	$6{\times}10^{-8}$	0.0099
1FS	0.0042	0.0055	0.0049	$5{\times}10^{-8}$	$8{\times}10^{-8}$	0.0232
2FS	$2{\times}10^{-7}$	0.0045	0.0057	$3{\times}10^{-7}$	$2{\times}10^{-7}$	0.0234
LFS	$2{\times}10^{-8}$	$1{\times}10^{-7}$	$4{\times}10^{-8}$	$5{\times}10^{-8}$	$6{\times}10^{-8}$	$5{\times}10^{-8}$

Table 1: Stress values for Euclidean inter-city distances.

	Australia	Canada	England	France	Türkiye	USA
Classical MDS	3787.0	3690.1	613.9	855.3	312.2	3639.0
ALSCAL	3786.9	3645.5	535.9	896.5	693.6	2932.5
NFS	1604.8	2739.2	408.0	457.0	787.2	2309.9
1FS	1413.8	1507.3	304.5	353.3	437.9	1651.5
2FS	774.0	297.2	168.7	152.6	172.2	554.0
LFS	$7{\times}10^{-5}$	0.0007	$2{\times}10^{-5}$	$5{\times}10^{-5}$	$7{\times}10^{-5}$	0.0001

Table 2: Location error values for Euclidean inter-city distances.

In this setting, we would like to mention that there is no accepted standard as to what value of the stress can be used as an indicator of a good representation. As a rule of thumb, we have resorted to the classification given by Kruskal: Any value less than 0.05 is excellent, values between 0.05 and 0.1 are satisfactory, and anything above 0.15 is unacceptable. From the results given in Table 1 we conclude that the performance of the new method is excellent.

Average location errors are presented in Table 2. The first two rows consist of location errors for classical MDS and ALSCAL. Since the results do not depend on the initial conditions, there is no need for repeating the runs in either of these methods. On the other hand, we report the average values of 10 or 100 replications depending on the strategy used for *NeuroMapRecon*. We observe that the location error decreases consistently for all data sets as the final configuration is more constrained, and goes practically to zero for the LFS.

Figure 1 illustrates the final configuration that is obtained for Türkiye data with LFS where the coordinates of two anchor cities are provided as additional data. In this case, the corresponding neurons are fixed to exactly fall on the coordinates of these cities. Each is shown with an " $\times$ ". Figure 2 displays the actual locations of the cities along with the points generated by ALSCAL. Observe the accuracy of *NeuroMapRecon* especially when it is armed with the line-fixed strategy.

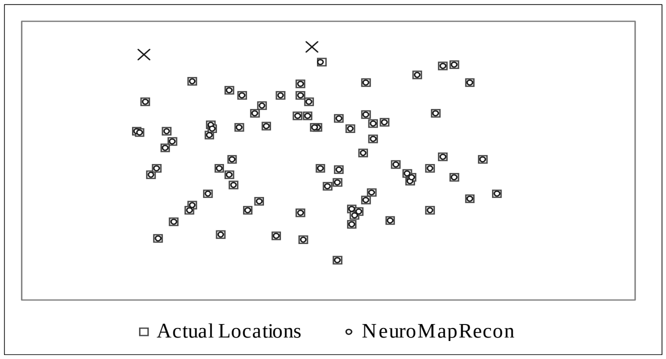

Figure 1: Final configuration obtained by *NeuroMapRecon* with line-fixed strategy for Türkiye data.

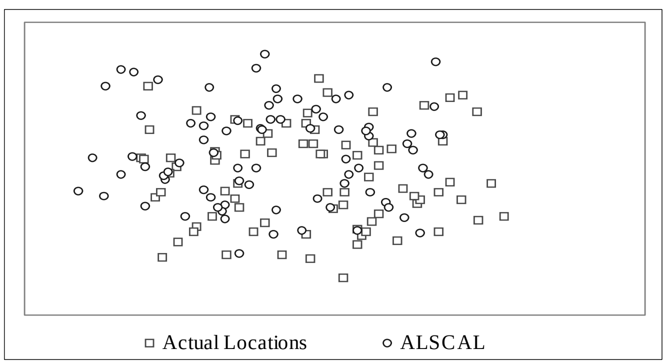

Figure 2: Final configuration obtained by ALSCAL for Türkiye data.

4.2 Results when Road Distances are not Euclidean

When the distance matrix consists of the road distances which are not Euclidean, the final configurations obtained by classical scaling and ALSCAL have nonzero stress values. It is for such real-life data settings that *NeuroMapRecon* demonstrates its true advantages. Stress values and location errors obtained with all three methods are given in Tables 3 and 4,

	Australia	Canada	England	France	Türkiye	USA
Classical MDS	0.0557	0.0252	0.0639	0.0345	0.0469	0.0419
ALSCAL	0.0509	0.0230	0.0600	0.0312	0.0422	0.0376
NFS	0.0452	0.0131	0.0444	0.0276	0.0390	0.0295
1FS	0.0467	0.0135	0.0452	0.0278	0.0391	0.0370
2FS	0.0551	0.0346	0.0513	0.0327	0.0452	0.0441
LFS	0.0494	0.0557	0.0740	0.0284	0.0462	0.0452

Table 3: Stress values for true inter-city distances.

	Australia	Canada	England	France	Türkiye	USA
Classical MDS	3101.6	2760.7	639.0	865.8	792.7	2936.8
ALSCAL	3896.0	3929.5	551.2	897.3	792.1	2935.8
NFS	1938.3	2869.2	452.3	471.9	805.5	2437.8
1FS	1710.4	1747.8	345.5	413.5	619.7	1693.8
2FS	1018.3	692.4	198.5	189.3	260.0	739.4
LFS	349.1	647.6	76.1	48.1	224.7	330.9

Table 4: Location error values for true inter-city distances.

respectively. The values in the rows for *NeuroMapRecon* are again the average values over 10 or 100 replications depending on the strategy. Based on the stress values reported in Table 3 we can say that ALSCAL performs better than classical scaling while *NeuroMapRecon* is the best method with respect to both performance measures. According to the stress scale proposed by Kruskal *NeuroMapRecon* seems to be "excellent". This is commendable especially when we realize that *NeuroMapRecon* is easy to implement and does not involve any matrix operations. Results in Table 3 indicate that in most of the cases, the stress increases as the final configuration is more constrained. This is not unexpected because imposing additional constraints to configurations may lead to unfavorable effect on the final stress values. For example, when we fix one of the neurons under 1FS, we remove one degree of freedom. Fixing two neurons instead of one, constrains the map further, and thereby results in an increase in the stress.

	5 missing δ_{ij}		10 missing δ_{ij}	
	NeuroMapRecon	ALSCAL	*NeuroMapRecon*	ALSCAL
Australia	0.04429	0.03898	0.04762	0.04805
Canada	0.01313	0.04126	0.01284	0.02126
England	0.04183	0.04130	0.04779	0.05788
France	0.03089	0.04068	0.02854	0.02902
Türkiye	0.03909	0.08008	0.03907	0.04212
USA	0.04460	0.04304	0.03626	0.03401

Table 5: Performance of *NeuroMapRecon* with missing input data.

4.3 Results for Incomplete Data Sets

NeuroMapRecon can also reconstruct maps even if some of the distance data are missing. In order to demonstrate the efficiency of the new method with respect to this property, experiments are conducted by removing some elements of the distance matrix, Δ. The "no-fixed" strategy is selected for this set of experiments as it gives the best results among the other strategies in terms of stress. In order to reduce the bias, for each data set the experiments are repeated 10 times by removing first 5 and then 10 δ_{ij}'s from Δ randomly. Since the final configuration given by *NeuroMapRecon* is dependent on the initial weights of the neurons, 10 replications are performed with random neuron initializations for each of the 10 runs. Note that a different set of distances are removed from Δ in each run. Therefore, the stress values reported in the first and third columns (*NeuroMapRecon* columns) of Table 5 are the averages of 100 replications. However, random initialization is not possible for ALSCAL and thus the numbers given in the second and fourth columns of Table 5 are the averages of 10 stress values.

Results obtained with *NeuroMapRecon* are again superior. Even in the presence of missing input data, it is capable of finding a final configuration with "excellent" stress values. ALSCAL performs slightly better for the Australia and England data with 5 missing values, and for the USA data with 5 and 10 missing values.

5 Conclusions

In this paper we have considered the fundamental problem of reconstructing a map (referred to as *MapRecon*) when the given data is the set of distances among cities in a region. This is the "inverse" of the distance estimation problem where the goal is to determine a good estimator for inter-city road distances as a function of given city coordinates. In the map reconstruction problem our aim is to determine the location of the cities in a two dimensional map such that the Euclidean distances among the points in the obtained configuration approximate true road distances as closely as possible. The reported solutions to this problem are few, and primarily involve traditional techniques used in MDS.

The solution we propose, called *NeuroMapRecon,* is very accurate and does not involve any intricate matrix computations or generalized inverse computations. It is also adaptive and can be said to be of a "real time" flavor. *NeuroMapRecon* has been rigorously tested on different data sets consisting of the road distances obtained from various countries by comparing the results with those provided by classical MDS method and ALSCAL. The accuracy of the proposed method is superior. *NeuroMapRecon* has also the following two desirable properties. First, it can reproduce configurations even if some of the input data are missing. Second, given the locations of at most two cities it is possible to obtain configurations without translation, rotation and reflection so that cities are located very close to their original locations (i.e., small location error). We are not aware of any other solution to *MapRecon* which possesses all these characteristics.

References

[1] İ.K. Altınel, N. Aras, J. Oommen, "Vector quantization for arbitrary distance function estimation," *Informs Journal on Computing*, vol. 9, pp. 439–451, 1997.

[2] J. Brimberg and R.F. Love, "Estimating distances," *Facility Location; A Survey of Applications and Methods*, Z. Drezner, Ed., Springer Series in Operations Research, 1995.

[3] R.F. Love and J.G. Morris, “Modelling inter-city road travel distances by mathematical functions,” *Opreational Research Quarterly*, vol. 23, pp. 61–71, 1972.

[4] R.F. Love and J.G. Morris, “Mathematical models of road travel distances,” *Management Sciences*, vol. 25, pp. 130–139, 1979.

[5] W.S. Torgerson, “Multidimensional scaling,” *Psychometrica*, vol. 17, pp. 401–419, 1952.

[6] I. Borg and P. Groenen, *Modern Multidimensional Scaling, Theory and Applications*, New York, Springer–Verlag, 1997.

[7] R.N. Shepard, “Analysis of proximities: Multidimensional scaling with an unknown distance function I & II,” *Psychometrica*, vol. 27, pp. 125–140, 219–246, 1962.

[8] W.S. Torgerson, “Multidimensional scaling of similarity,” *Psychometrica*, 30, 379–393, 1965.

[9] C. Chatfield, *Introduction to Multivariate Analysis*, Chapman and Hall, 1980.

[10] M. Steyvers, “Multidimensional scaling” in *Encyclopedia of Cognitive Science*, 2002.

[11] Y. Takane, F.W. Young and J. de Leeuw, “Nonmetric individual differences multidimensional scaling: An alternating least squares method with optimal scaling features,” *Psychometrika*, vol. 42, pp. 7–67, 1977.

[12] D.H. Gray, “Vector Quantization,” *IEEE ASSP Magazine*, vol. 1, pp. 4–29, 1984.

[13] Y. Linde, A. Buzo and R.M. Gray, “An algorithm for vector quantization,” *IEEE Transactions on Communications*, COM–28, pp. 61–71, 1980.

[14] T. Kohonen, “The self-organizing map,” *Proc. IEEE*, vol. 78, pp. 1464–1480, 1990.

[15] T. Kohonen, *The self-organizing map*, Berlin, Germany: Springer-Verlag, 1995.

[16] R. P. W. Duin and E. Pekalska, “Possibilities of Zero-Error Recognition by Dissimilarity Representations,” in *Proc. of PRIS-2002, The 2002 International Workshop on Pattern Recognition in Information Systems*, Alicante, 2002, pp. 20–32.

[17] E. Pekalska and R. P. W. Duin, 'Classifiers for Dissimilarity-based Pattern Recognition', *Proceedings of ICPR-2000, The International Conference on Pattern Recognition*, **2**, 12–16, 2000.

[18] E. Pekalska and R. P. W. Duin, "Automatic Pattern Recognition by Similarity Representations," *Electronic Letters*, vol. 37, pp. 159–160, 2001.

[19] J.B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrica*, vol. 29, pp. 1–27, 1964.

[20] M. Steyvers and T. Busey, "Predicting similarity ratings to faces using physical descriptions" in *Computational, geometric, and process perspectives on facial cognition: Contexts and challenges*, M. Wenger and J. Townsend Eds., Lawrence Erlbaum Associates, 2000.

[21] R.F. Love and J.H. Walker, "Distance data for eighteen geographic regions," Working Paper #383, Michael G. DeGroote School of Business, McMaster University, 1993.