Non-Linear Dimensionality Reduction with Input Distances Preservation

Lluís Garrido[(1)], Sergio Gómez[(2)], Jaume Roca[(1)]

[(1)]Dept. d'Estructura i Constituents de la Matèria/IFAE,
Universitat de Barcelona,
Diagonal 647,
E-08028 Barcelona, Spain.
[(2)]Dept. d'Enginyeria Informàtica i Matemàtiques (ETSE),
Universitat Rovira i Virgili,
Crt. de Salou s/n,
E-43006 Tarragona, Spain.

Appeared in *Proceedings of the Ninth International Conference on Artificial Neural Networks*, Published by the Institution of Electrical Engineers, Vol. 2, London (1999) 922–927.

Abstract

A new error term for dimensionality reduction, which clearly improves the quality of NLPCA neural networks, is introduced, and some illustrative examples are given. The method tries to maintain the original data structure by preserving the distances between data points.

1 Introduction

One of the standard problems in the analysis of multidimensional data is that of dimensionality reduction, i.e. the elaboration of a map $\varphi : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ which transforms a collection of n-dimensional points $\boldsymbol{x}_a \in \mathbb{R}^n$ $(a = 1, \dots, p)$ to a lower dimensional space $(m < n)$, in such a way that the distribution of the projected points $\varphi(\boldsymbol{x}_a)$ is similar to that of the original set. Tipically, m is chosen to be two or three in order to make available a graphical representation of the projected configuration. This can help visualize an underlying structure that might be obscured by the cluttering of data in the original space.

Principal Component Analysis (PCA) stands out among the most popular dimensionality reduction methods. Its aim is to find the linear projection that accounts for as much as possible of the data's variance. Several learning rules, based either on hebbian learning or back-propagation, have been proposed to train simple feed-forward neural networks to perform PCA (see e.g. [8, 9]). The generalization of these methods to more sophisticated architectures of the nets gives rise to different versions of Non-Linear PCA (NLPCA) [2, 3, 4, 6, 7], which get rid of the unnecessary linearity constraint of PCA.

A typical implementation of NLPCA makes use of a multilayer neural network with sigmoidal units and several hidden layers, one of which has a small number of units and is known as the bottle-neck layer, as shown in Fig. 1. This net is trained through self-supervised back-propagation (i.e. inputs considered as well as desired outputs), and the projection mapping φ is finally read as the part of the network going from the input to the bottle-neck layer.

The main problem of NLPCA is its excessive freedom in the selection of the bottle-neck representation of the data: since the only training condition is the minimization of the quadratic error at the output layer, the net can choose any bottle-neck distribution which allows a satisfactory approximate inversion of φ, no matter if it resembles or

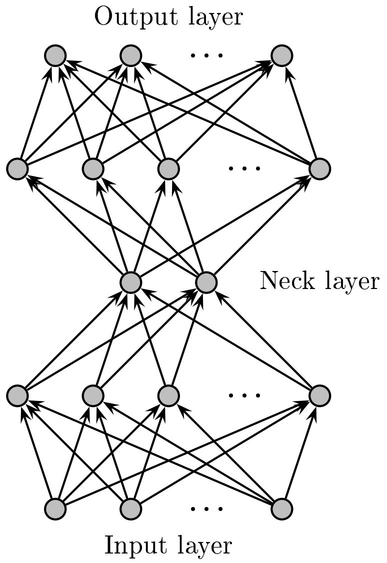

Figure 1: Architecture of NLPCA neural networks.

not to the original distribution.

In this article we review our technique exposed in [5], and provide two examples which show the improvement obtained over PCA and NLPCA. Our main idea is borrowed from Multidimensional Scaling (MDS), a set of methods to assign coordinates to points when only their relative mutual distances are known beforehand [1]. However, the standard approach to MDS, known as Classical Scaling, also reduces to PCA once the coordinates of the points are already known.

2 The method

Let $d_{ab}^{(n)}$ be the euclidean distance between patterns a and b,

$$d_{ab}^{(n)} = \sqrt{(\boldsymbol{x}_a - \boldsymbol{x}_b)^2}, \qquad (1)$$

and let $d_{ab}^{(m)}$ be their euclidean distance in the projected space,

$$d_{ab}^{(m)} = \sqrt{(\varphi(\boldsymbol{x}_a) - \varphi(\boldsymbol{x}_b))^2}. \qquad (2)$$

If we want to impose that the sets $\{\boldsymbol{x}_a\}$ and $\{\varphi(\boldsymbol{x}_a)\}$ approximately share the same distribution, a suitable condition could be the minimization of

$$E_\varphi = \frac{1}{2}\sum_{a,b}\left(d_{ab}^{(n)} - d_{ab}^{(m)}\right)^2. \qquad (3)$$

When φ is restricted to linear functions, it can be shown that the minimization of E_φ reduces to PCA. Thus, Eq. (3) constitutes a good starting point for a non-linear generalization of PCA.

Our projection method consists basically in the use of the neural network implementation of NLPCA explained above, properly modified so as to minimize E_φ. However, several additional aspects have to be explained to deal with the bounded nature of sigmoids.

It is convenient to perform first a translation and a global scaling of the initial data,

$$\boldsymbol{x}_a \longrightarrow \boldsymbol{\xi}_a^{\text{in}} = \lambda^{\text{in}}(\boldsymbol{x}_a - \boldsymbol{a}), \qquad (4)$$

to make $\boldsymbol{\xi}_a^{\text{in}} \in [0,1]^n$, where $[0,1]$ is the interval we have chosen for the outcome of the sigmoids. Let us call $\boldsymbol{\xi}_a^{\text{out}} \in [0,1]^n$ and $\boldsymbol{\xi}_a^{\text{nk}} \in [0,1]^m$ respectively the output and the neck layer activation when $\boldsymbol{\xi}_a^{\text{in}}$ is presented to the net. Thus, it is satisfied that $0 \leq d_{ab}^{\text{in}} \leq \sqrt{n}$ and $0 \leq d_{ab}^{\text{nk}} \leq \sqrt{m}$, which means Eq. (3) is not directly applicable.

The error function we propose for the back-propagation method is

$$E_{\text{NNDP}} = \alpha\ E_1 + (1-\alpha)\ E_2\,, \quad \alpha \in [0,1] \qquad (5)$$

where

$$E_1 = \sum_a \left(\boldsymbol{\xi}_a^{\text{out}} - \boldsymbol{\xi}_a^{\text{in}}\right)^2 \qquad (6)$$

favours those maps for which the representation in the bottle-neck layer can be best accurately inverted to recover the original configuration, and

$$E_2 = \sum_{a,b}\left(\frac{d_{ab}^{\text{in}}}{\sqrt{n}} - \frac{d_{ab}^{\text{nk}}}{\sqrt{m}}\right)^2, \qquad (7)$$

forces the representation in the bottle-neck to inherit, as closely as possible, the metric structure of the original configuration. E_2 departs from E_φ in the different scalings of d_{ab}^{in} and d_{ab}^{nk}, which have been introduced to have them in the same range.

The various scalings involved in this process make the outcome of the neck layer not to be directly interpretable as the final answer; we can bring it back to the original scale by setting

$$\varphi_{\text{NNDP}}(\boldsymbol{x}_a) = \lambda^{\text{out}}\boldsymbol{\xi}_a^{\text{nk}}, \qquad (8)$$

with $\lambda^{\rm out} = \sqrt{n/m}\ \lambda^{\rm in}$. However, a slightly better solution can be obtained by choosing instead

$$\lambda^{\rm out} = \frac{\sum_{a,b} d_{ab}^{(n)} d_{ab}^{\rm nk}}{\sum_{a,b} \left(d_{ab}^{\rm nk}\right)^2}, \qquad (9)$$

since this is the value of λ that minimizes the function $E(\lambda) = \frac{1}{2}\sum_{a,b}(d_{ab}^{(n)} - \lambda d_{ab}^{\rm nk})^2$ for the given neck configuration, which is what we are ultimately trying to achieve with the whole procedure.

In the practical use of the neural network we have noticed that the best results are obtained by letting the parameter α fall to zero as the learning grows so that the error function $E_{\rm NNDP}$ reduces to E_2 after a certain number of iterations. Actually, a non-zero value of α is only useful in the early stages of the learning, in order to speed up convergence. In this situation, i.e. with $E_{\rm NNDP} = E_2$, it is easy to prove analytically that the configuration minimizing $E_{\rm NNDP}$ differs from the one minimizing E_φ only by a global scaling $\sqrt{n/m}$ of all coordinates. Thus, the (otherwise technically convenient) scalings that we have introduced above are completely harmless for the purpose of searching for the best mapped configuration.

3 Two benchmarks

To show the benefits of our method we have applied it to two simple examples having $n = 3$ and $m = 2$, comparing our NNDP results with those obtained by PCA and standard NLPCA. To measure the overall relative distance error we will use

$$\varepsilon = \sqrt{\frac{\sum_{a,b}\left(d_{ab}^{(n)} - d_{ab}^{(m)}\right)^2}{\sum_{a,b}\left(d_{ab}^{(n)}\right)^2}}. \qquad (10)$$

The first example consists in a set of points which form the twisted band of Fig. 2. They are 85 (17×5) points laying on equally spaced 0.8-long segments (5 points per segment, spacing of 0.2). The first segment is at $y = 0.1$ and the last one at $y = 0.9$. Segments rotate with constant speed, so that the first one is parallel to the x axis, and so is the last one, with a π rotation between them. The whole figure fits in the cube $[0.1, 0.9]^3$.

When PCA is applied to it, the result is that of Fig. 3, which displays an artificial singular point absent in the original data. A typical run of NLPCA, in Fig. 4, shows its ability to avoid the ficticious singular point, by adapting the projection to the geometry of the band. However, our NNDP projection, in Fig. 5, is even capable of keeping the relative distances between most of the points. To show the highly non-linear nature of this mapping we have included in Figs. 6, 7 and 8 the projections of nine reference planes which are parallel to the main coordinate planes.

The errors obtained for each method are

$$\begin{aligned} \varepsilon_{PCA} &= 0.2434\,, \\ \varepsilon_{NLPCA} &= 0.2451\,, \\ \varepsilon_{NNDP} &= 0.1722\,. \end{aligned}$$

The second example consists in two orthogonally chained elliptic rings as shown in Fig. 9, so that the whole figure fits exactly in the $[0.1, 0.9]^3$ cube. Each ring contains 41 points, crossing the other one at its center.

Now, PCA, in Fig. 10, perfectly projects one of the rings, but the other one degenerates to a line. Again, NLPCA is much better as it avoids degeneration, Fig. 11. However, the lack of an objective function for the bottle-neck produces a large variety of projected shapes, which only have in common the topology (two closed curves intersecting in two points), and with errors covering a wide range. NNDP generates more coherent configurations, all of them similar to the one in Fig. 12, in which the simmetry of the original data is clearly manifested. This projection is not very far from being a linear one, as reflected by the projection of the nine reference planes shown in Figs. 13, 14 and 15, yet it could not be obtained by PCA.

In this example, the errors obtained for each method are

$$\begin{aligned} \varepsilon_{PCA} &= 0.3024\,, \\ \varepsilon_{NLPCA} &= 0.2915\,, \\ \varepsilon_{NNDP} &= 0.1950\,. \end{aligned}$$

All the errors computed in this section correspond to the projections obtained once the optimal scaling of Eq. (9) has been performed, whereas the plots are shown unscaled.

References

[1] T.F. Cox and M.A.A. Cox, *Multidimensional Scaling*. Chapman & Hall, London 1994.

[2] D. DeMers and G. Cottrell 1994. Non-Linear Dimensionality Reduction, Published in *NIPS* **5**.

[3] Ll. Garrido, V. Gaitán, M. Serra-Ricart and X. Calbet 1995. Use of multilayer feedforward neural nets as a display method for multidimensional distributions, *Int. J. Neural Systems* **6**, 273.

[4] Ll. Garrido, S. Gómez, V. Gaitán and M. Serra-Ricart 1996. A regularization term to avoid the saturation of the sigmoids in multilayer neural networks, *Int. J. Neural Systems* **7**, 257.

[5] Ll. Garrido, S. Gómez and J. Roca 1999. Improved Multidimensional Scaling Analysis Using Neural Networks with Distance-Error Backpropagation, *Neural Computation* **11**, 595.

[6] N. Kambhatla and T.K. Leen 1995. Fast Non-Linear Dimension Reduction, Published in *NIPS* **6**.

[7] M.A. Kramer 1991. Non-linear principal component analysis using autoassociative neural networks, *AICHE Journal* **37**, 233.

[8] E. Oja 1982. A simplified neuron model as a principal component analyzer, *J. Math. Biol.* **15**, 267.

[9] T.D. Sanger 1989. Optimal unsupervised learning in a single-layer linear feedforward neural network, *Neural Networks* **2**, 459.

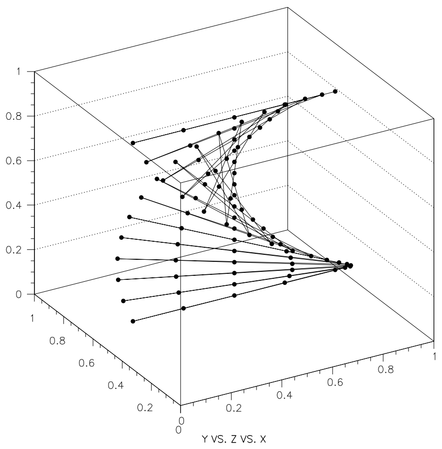

Figure 2: Twisted band.

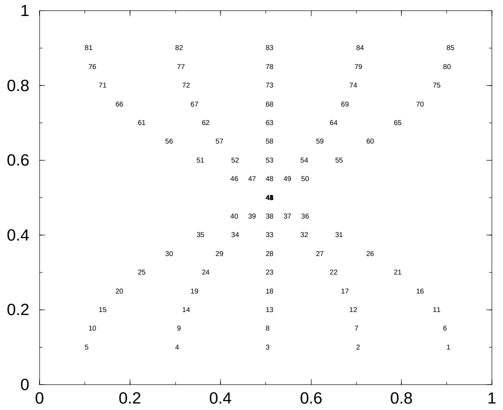

Figure 3: PCA of the twisted band.

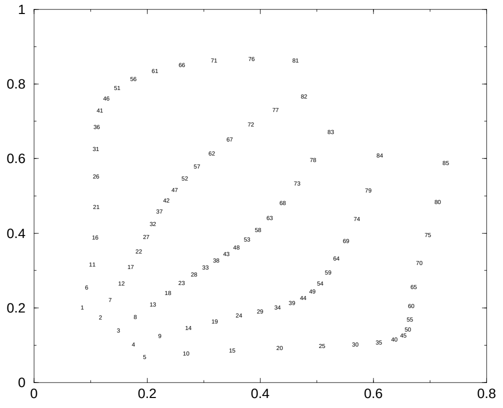

Figure 4: NLPCA of the twisted band.

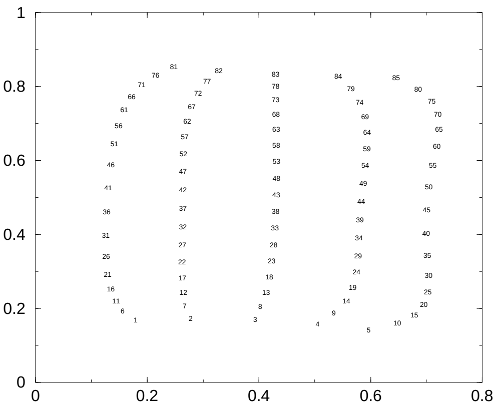

Figure 5: NNDP of the twisted band.

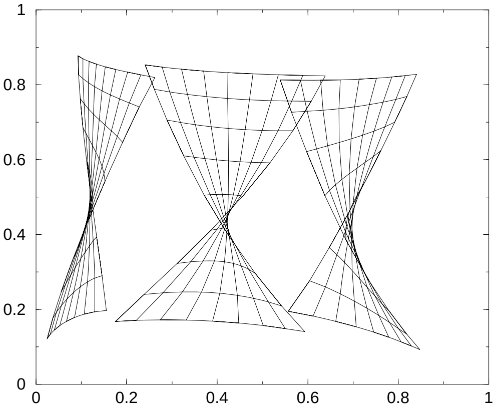

Figure 6: NNDP of XY planes of the twisted band.

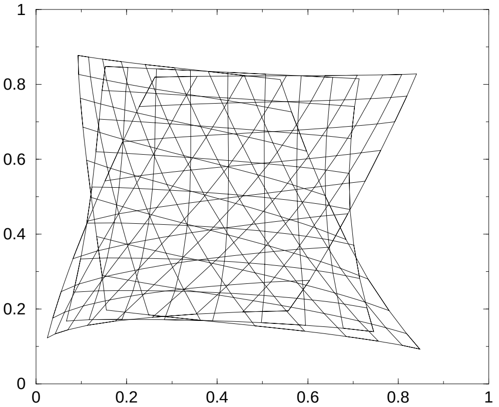

Figure 7: NNDP of YZ planes of the twisted band.

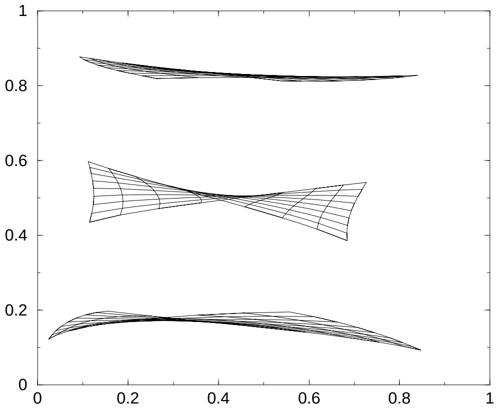

Figure 8: NNDP of ZX planes of the twisted band.

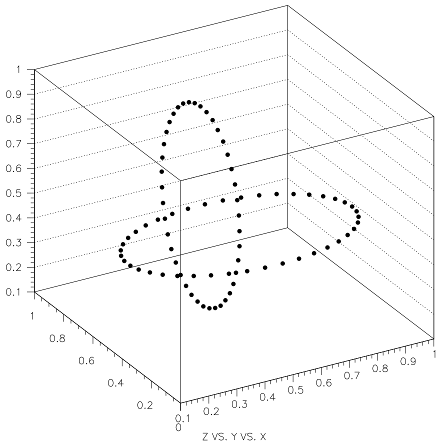

Figure 9: Chained rings.

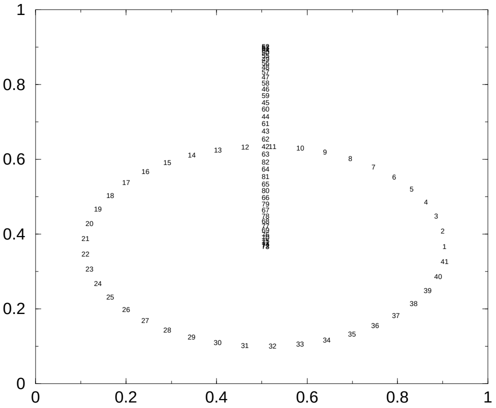

Figure 10: PCA of the chained rings.

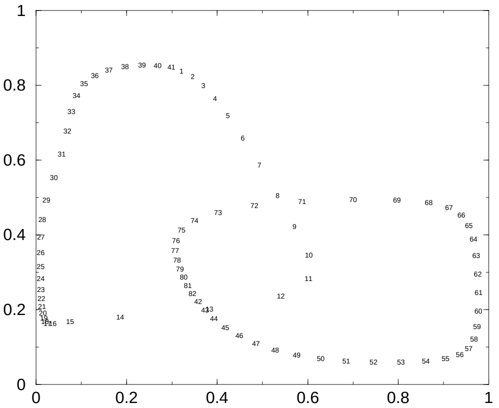

Figure 11: NLPCA of the chained rings.

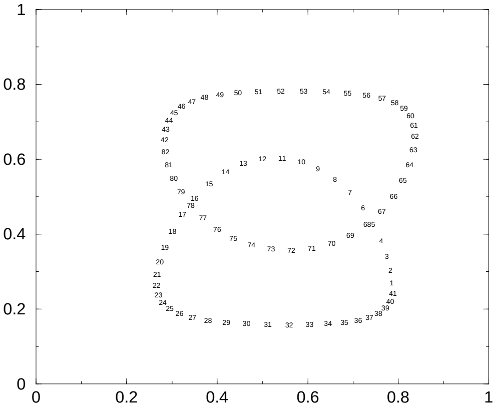

Figure 12: NNDP of the chained rings.

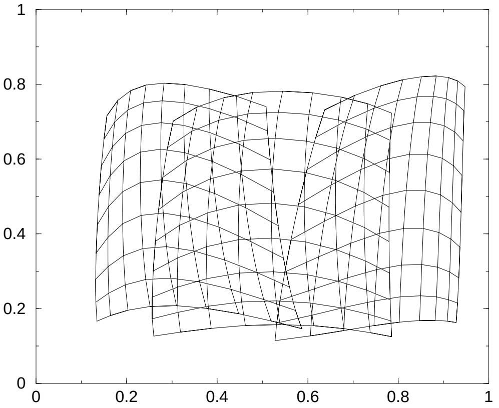

Figure 13: NNDP of XY planes of the chained rings.

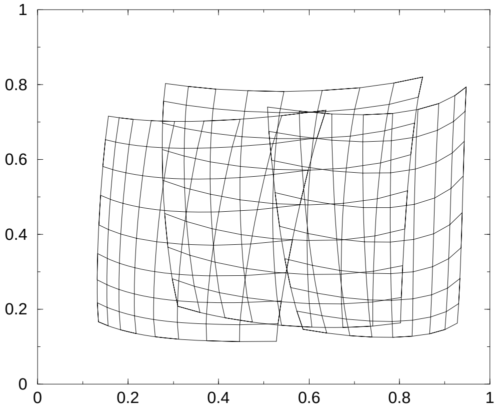

Figure 14: NNDP of YZ planes of the chained rings.

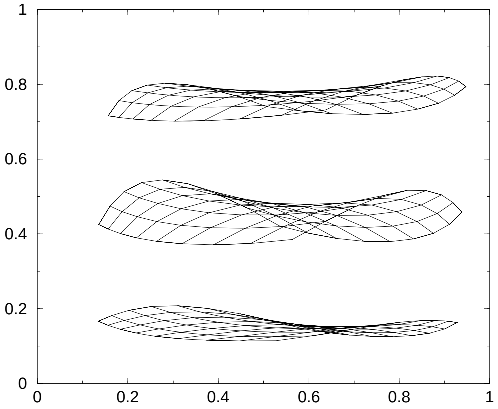

Figure 15: NNDP of ZX planes of the chained rings.