Tellus (2001), 53A, 599–615
Printed in UK. All rights reserved

TELLUS
ISSN 0280–6495

Nonlinear principal component analysis by neural networks

By WILLIAM W. HSIEH*, *Oceanography/EOS, University of British Columbia, Vancouver, BC, V6T1Z4, Canada*

(Manuscript received 27 September 2000; in final form 5 March 2001)

ABSTRACT

Nonlinear principal component analysis (NLPCA) can be performed by a neural network model which nonlinearly generalizes the classical principal component analysis (PCA) method. The presence of local minima in the cost function renders the NLPCA somewhat unstable, as optimizations started from different initial parameters often converge to different minima. Regularization by adding weight penalty terms to the cost function is shown to improve the stability of the NLPCA. With the linear approach, there is a dichotomy between PCA and rotated PCA methods, as it is generally impossible to have a solution simultaneously (a) explaining maximum global variance of the data, and (b) approaching local data clusters. With the NLPCA, both objectives (a) and (b) can be attained together, thus the nonlinearity in NLPCA unifies the PCA and rotated PCA approaches. With a circular node at the network bottleneck, the NLPCA is able to extract periodic or wave modes. The Lorenz (1963) 3-component chaotic system and the monthly tropical Pacific sea surface temperatures (1950-1999) are used to illustrated the NLPCA approach.

1. Introduction

Having to analyze large fields of data, from satellite images to numerical model output, meteorologists and oceanographers have embraced classical multivariate statistical methods, such as principal component analysis (PCA) and canonical correlation analysis (CCA) (Von Storch and Zwiers, 1999). PCA (also known as empirical orthogonal function analysis) extracts the modes in a set of variables $\{x_i\}$. It is commonly used for two purposes: (i) to reduce the dimensionality of the dataset by retaining only the first few modes, and (ii) to extract features (or recognize patterns) from $\{x_i\}$ — a task at which it is challenged by rotated PCA (RPCA) methods (Richman, 1986).

While much has been learned through the use of PCA and related methods, the fact that they are linear methods implies a potential oversimplification of the datasets being analyzed. The advent of neural network (NN) models, a class of powerful nonlinear empirical modelling methods originating from the field of artificial intelligence, raises the hope that the linear restriction in our analysis of environmental datasets may finally be lifted (Hsieh and Tang, 1998).

Various NN methods have been developed for performing PCA (Oja, 1982; Diamantaras and Kung, 1996). Nonlinear principal component analysis (NLPCA) using NN was first introduced by Kramer (1991) in the chemical engineering literature, and is now used by researchers in many fields. Due to the presence of multiple minima in the cost function, the NLPCA is generally less stable than its linear counterpart. The first objective of this paper is to illustrate how the stability of the NLPCA can be improved by adding weight penalty terms to the cost function.

The tropical Pacific sea surface temperature (SST) and sea level pressure fields have recently been analyzed by the NLPCA (Monahan, 2001). Although comparisons have been made between

* e-mail: whsieh@eos.ubc.ca

the NLPCA and PCA methods in Monahan (2001), the second objective of this paper is to examine the role of NLPCA in a broader context, in particular its relation to the RPCA methods, as well as to the PCA method, and show why the dichotomy between PCA and RPCA resolves automatically with the introduction of nonlinearity in NLPCA.

PCA and RPCA are known to handle data containing periodic phenomena or waves rather poorly. The third objective of this paper is to show that NLPCA with a circular node in the network bottleneck (Kirby and Miranda, 1996) generalizes the NLPCA to handle periodic or wave phenomena as well.

This paper is organized as follows: The theory of the NLPCA is given in Section 2. In Section 3, weight penalty terms are used to improve the stability of the NLPCA. A 3-way comparison between NLPCA, RPCA and PCA is performed on the tropical Pacific SST field in Section 4. The circular-noded NLPCA is presented and applied to the tropical Pacific SST in Section 5.

2. Theory of NLPCA

In most meteorological/oceanographic applications, the data can be expressed in the form $\boldsymbol{x}(t) = [x_1, ..., x_l]$, where each variable x_i, $(i = 1, ..., l)$, is a time series containing n observations. PCA looks for u, a linear combination of the x_i, and an associated vector $\boldsymbol{a}$, with

$$u(t) = \boldsymbol{a} \cdot \boldsymbol{x}(t), \tag{1}$$

so that

$$\langle \| \boldsymbol{x}(t) - \boldsymbol{a}u(t) \|^2 \rangle \quad \text{is minimized}, \tag{2}$$

where $\langle \cdots \rangle$ denotes a sample or time mean. Here u, called the first principal component (PC), is a time series, while $\boldsymbol{a}$, the first eigenvector of the data covariance matrix, (also called an empirical orthogonal function, EOF), often describes a spatial pattern. From the residual, $\boldsymbol{x} - \boldsymbol{a}u$, the second PCA mode can similarly be extracted, and so on for the higher modes. In practice, the common algorithms for PCA extract all modes simultaneously (Jollife, 1986; Preisendorfer, 1988).

The fundamental difference between NLPCA and PCA is that NLPCA allows a nonlinear mapping from $\boldsymbol{x}$ to u whereas PCA only allows a linear mapping. To perform NLPCA, the NN in Fig. 1 contains 3 "hidden" layers of variables (or "neurons") between the input and output layers of variables. (For the reader not familiar with NN models, see Hsieh and Tang, 1998). A transfer function f_1 maps from $\boldsymbol{x}$, the input column vector of length l, to the first hidden layer (the encoding layer), represented by $\boldsymbol{h}^{(x)}$, a column vector of length m, with elements

$$h_k^{(x)} = f_1[(\boldsymbol{W}^{(x)}\boldsymbol{x} + \boldsymbol{b}^{(x)})_k], \tag{3}$$

where (with the capital bold font reserved for matrices and the small bold font for vectors), $\boldsymbol{W}^{(x)}$ is an $m \times l$ weight matrix, $\boldsymbol{b}^{(x)}$, a column vector of length m containing the bias parameters, and $k = 1, ..., m$. Similarly, a second transfer function f_2 maps from the encoding layer to the bottleneck layer containing a single neuron, which represents the nonlinear principal component u,

$$u = f_2(\boldsymbol{w}^{(x)} \cdot \boldsymbol{h}^{(x)} + \bar{b}^{(x)}). \tag{4}$$

The transfer function f_1 is generally nonlinear (usually the hyperbolic tangent or the sigmoidal function, though the exact form is not critical), while f_2 is usually taken to be the identity function.

Next, a transfer function f_3 maps from u to the

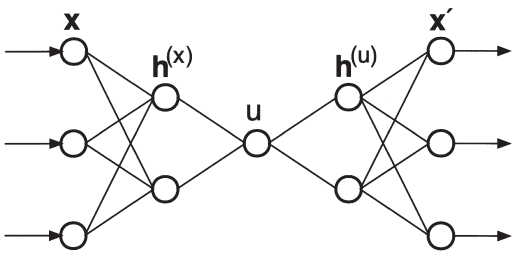

Fig. 1. The NN model for calculating nonlinear PCA (NLPCA). There are 3 "hidden" layers of variables or "neurons" (denoted by circles) sandwiched between the input layer $\boldsymbol{x}$ on the left and the output layer $\boldsymbol{x}'$ on the right. Next to the input layer is the encoding layer, followed by the "bottleneck" layer (with a single neuron u), which is then followed by the decoding layer. A nonlinear function maps from the higher dimension input space to the lower dimension bottleneck space, followed by an inverse transform mapping from the bottleneck space back to the original space represented by the outputs, which are to be as close to the inputs as possible by minimizing the cost function $J = \langle \| \boldsymbol{x} - \boldsymbol{x}' \|^2 \rangle$. Data compression is achieved by the bottleneck, with the bottleneck neuron giving u, the nonlinear principal component.

final hidden layer (the decoding layer) $\boldsymbol{h}^{(u)}$,

$$h_k^{(u)} = f_3[(\boldsymbol{w}^{(u)}u + \boldsymbol{b}^{(u)})_k], \tag{5}$$

($k = 1, \ldots, m$); followed by f_4 mapping from $\boldsymbol{h}^{(u)}$ to $\boldsymbol{x}'$, the output column vector of length l, with

$$x_i' = f_4[(\boldsymbol{W}^{(u)}\boldsymbol{h}^{(u)} + \bar{\boldsymbol{b}}^{(u)})_i]. \tag{6}$$

The cost function $J = \langle \| \boldsymbol{x} - \boldsymbol{x}' \|^2 \rangle$ is minimized by finding the optimal values of $\boldsymbol{W}^{(x)}$, $\boldsymbol{b}^{(x)}$, $\boldsymbol{w}^{(x)}$, $\bar{\boldsymbol{b}}^{(x)}$, $\boldsymbol{w}^{(u)}$, $\boldsymbol{b}^{(u)}$, $\boldsymbol{W}^{(u)}$ and $\bar{\boldsymbol{b}}^{(u)}$. The MSE (mean square error) between the NN output $\boldsymbol{x}'$ and the original data $\boldsymbol{x}$ is thus minimized. The NLPCA was implemented using the hyperbolic tangent function for f_1 and f_3, and the identity function for f_2 and f_4, so that

$$u = \boldsymbol{w}^{(x)} \cdot \boldsymbol{h}^{(x)} + \bar{b}^{(x)}, \tag{7}$$

$$x_i' = (\boldsymbol{W}^{(u)}\boldsymbol{h}^{(u)} + \bar{\boldsymbol{b}}^{(u)})_i, \tag{8}$$

Without loss of generality, we impose the constraint $\langle u \rangle = 0$, hence

$$\bar{b}^{(x)} = -\langle \boldsymbol{w}^{(x)} \cdot \boldsymbol{h}^{(x)} \rangle. \tag{9}$$

The total number of free (weight and bias) parameters used by the NLPCA is then $2lm + 4m + l$. Furthermore, we adopt the normalization condition that $\langle u^2 \rangle = 1$. This condition is approximately satisfied by modifying the cost function to

$$J = \langle \| \boldsymbol{x} - \boldsymbol{x}' \|^2 \rangle + (\langle u^2 \rangle - 1)^2. \tag{10}$$

The choice of m, the number of hidden neurons in both the encoding and decoding layers, follows a general principle of parsimony. A larger m increases the nonlinear modelling capability of the network, but could also lead to overfitted solutions (i.e., wiggly solutions which fit to the noise in the data). If f_4 is the identity function, and $m = 1$, then (8) implies that all x_i' are linearly related to a single hidden neuron, hence there can only be a linear relation between the x_i' variables. For nonlinear solutions, we need to look at $m \geqslant 2$.

In effect, the linear relation (1) is now generalized to $u = f(\boldsymbol{x})$, where f can be any nonlinear function representable by a feed-forward NN mapping from the input layer to the bottleneck layer; and instead of (2), $\langle \| \boldsymbol{x} - \boldsymbol{g}(u) \|^2 \rangle$ is minimized, where $\boldsymbol{g}$ is the generally nonlinear function mapping from the bottleneck to the output layer. The residual, $\boldsymbol{x} - \boldsymbol{g}(u)$, can be input into the same network to extract the second NLPCA mode, and so on for the higher modes.

The nonlinear optimization was carried out by the MATLAB function "fminu", a quasi-Newton algorithm. Because of local minima in the cost function, there is no guarantee that the optimization algorithm reaches the global minimum. Hence an ensemble of 30 NNs with random initial weights and bias parameters was run. Also, 20% of the data was randomly selected as test data and withheld from the training of the NNs. Runs where the MSE was larger for the test dataset than for the training dataset were rejected to avoid overfitted solutions. Then the NN with the smallest MSE was selected as the solution.

For the nonlinear optimization to work well, appropriate scaling of the $\boldsymbol{x}$ variables is needed. Suppose an NLPCA model has been successfully developed for the data $\boldsymbol{x}$, yielding output $\boldsymbol{x}'$. We now want to test the effect of scaling all the input variables by a factor α, i.e., $\boldsymbol{x}$ is replaced by $\alpha\boldsymbol{x}$. To get $\alpha\boldsymbol{x}'$ as the output, only $\boldsymbol{W}^{(x)}$ needs to be replaced by $\boldsymbol{W}^{(x)}/\alpha$ in (3), and $\boldsymbol{W}^{(u)}$ by $\alpha\boldsymbol{W}^{(u)}$ in (8), with all other parameters and hidden neurons unchanged. Suppose the elements of $\boldsymbol{W}^{(x)}$ and $\boldsymbol{W}^{(u)}$ are of the same order of magnitude, if α is quite different from order 1, then the elements of $\boldsymbol{W}^{(x)}/\alpha$ and $\alpha\boldsymbol{W}^{(u)}$ will have very different magnitudes. The nonlinear optimization algorithm does not work well if the parameters to be determined have a wide range of magnitudes. One possibility is to standardize all the input variables, i.e., for each variable, remove its mean and divide by its standard deviation. If the input variables are themselves the leading PCs (i.e., PCA has been used to compact the dataset), then standardization would exaggerate the importance of the higher PCA modes. In this situation, it would be appropriate to normalize each input variable by subtracting its mean and dividing by the standard deviation of the first PC.

That the classical PCA is indeed a linear version of this NLPCA can be readily seen by replacing all the transfer functions with the identity function, thereby removing the nonlinear modelling capability of the NLPCA. Then the forward map to u involves only a linear combination of the original variables as in the PCA.

In Fig. 1, only a single hidden layer is used in the mapping from the input layer to the bottleneck, and also in the mapping from the bottleneck to the output layer, since given enough hidden neurons, any continuous function can be approximated to arbitrary accuracy by one hidden layer

(Cybenko, 1989). The NLPCA here generalizes easily to more than one hidden layer mappings, as two hidden-layer mappings may outperform single hidden layer mappings in modelling complicated nonlinear functions.

It is possible to have more than one neuron at the bottleneck layer. For instance, with two bottleneck neurons, the mode extracted will span a 2-D surface instead of a 1-D curve. Such higher-dimensional modes are generally more difficult to visualize and will not be pursued here.

3. Weight penalty

In general, the most serious problem with NLPCA is the presence of local minima in the cost function. As a result, optimizations started from different initial parameters often converge to different minima, rendering the method unstable. As an example, consider the famous Lorenz "butterfly"-shaped attractor from chaos theory (Lorenz, 1963). Describing idealized atmospheric convection, the Lorenz system is governed by 3 (nondimensionalized) differential equations:

$$\dot{x}_1 = -ax_1 + ax_2, \qquad \dot{x}_2 = -x_1x_3 + bx_1 - x_2,$$
$$\dot{x}_3 = x_1x_2 - cx_3, \tag{11}$$

where the overhead dot denotes a time derivative, and a, b and c are 3 parameters. A chaotic system is generated by choosing $a = 10$, $b = 28$, and $c = 8/3$. Fig. 2 illustrates the butterfly-shaped attractor produced by a dataset containing 1000 data points.

With x_1, x_2 and x_3 as inputs to the NLPCA network, the first mode extracted has a Z-shaped appearance (Fig. 2), in sharp contrast to the smooth U-shaped solution (Fig. 3) found by Monahan (2000). Why is there such a drastic difference? If one starts with small random initial parameters, and terminate the optimization algorithm after a relative small number of iterations, one arrives at the U-shaped solution of Monahan. However, with larger random initial weights and/or more iterations, the deeper minimum is reached, which gives the Z-shaped solution. For comparison, the first PCA mode is the straight line shown in Figs. 2, 3.

More precisely, the shape in Figs. 2b,c is a mirror image of Z — in fact, the solution can be either Z-shaped or mirror Z-shaped. The Lorenz system is invariant to the transformation

$$(x_1, x_2, x_3) \rightarrow (-x_1, -x_2, x_3). \tag{12}$$

If one applies this transformation to the solution in Fig. 2, one will find a Z-shaped solution in Figs. 2b, c. Of course, with a finite dataset, one does not have the perfect symmetry of the differential equations, so either the minimum associated with the Z-shape or that with the mirror Z-shape will be the deeper minimum. This means that depending on the sample, or the training datasets selected, one can get either the Z solution or the mirror Z solution — a very unstable and undesirable outcome.

The U-shaped solution is not only esthetically more pleasant than the Z or mirror Z-shaped solutions, but is also invariant with respect to the transformation (12). How does one arrive at the U-shaped solution without worrying about whether the initial random weights were small enough and whether the number of iterations used was small enough? Regularization of the cost function by adding weight penalty terms is an answer.

The purpose of the weight penalty terms is to limit the nonlinear power of the NLPCA, which came from the nonlinear transfer functions in the network. The transfer function tanh has the property that given x in the interval $[-L, L]$, one can find a small enough weight w, so that $\tanh(wx) \approx wx$, i.e., the transfer function is almost linear. Similarly, one can choose a large enough w, so that tanh approaches a step function, thus yielding Z-shaped solutions. If we can penalize the use of excessive weights, we can limit the degree of nonlinearity in the NLPCA solution. This is achieved with a modified cost function

$$J = \langle \| \boldsymbol{x} - \boldsymbol{x}' \|^2 \rangle + (\langle u^2 \rangle - 1)^2 + p \sum_{ki} (W_{ki}^{(x)})^2, \tag{13}$$

where p is the weight penalty parameter. A large p increases the concavity of the cost function, and forces the weights $\boldsymbol{W}^{(x)}$ to be small in magnitude, thereby yielding smoother and less nonlinear solutions than when p is small or zero. With a large enough p, the danger of overfitting is greatly reduced, hence the optimization can proceed until convergence to the global minimum. Of course, if p is too large, one gets only the linear solution — and ultimately the trivial solution (where all weights are zero).

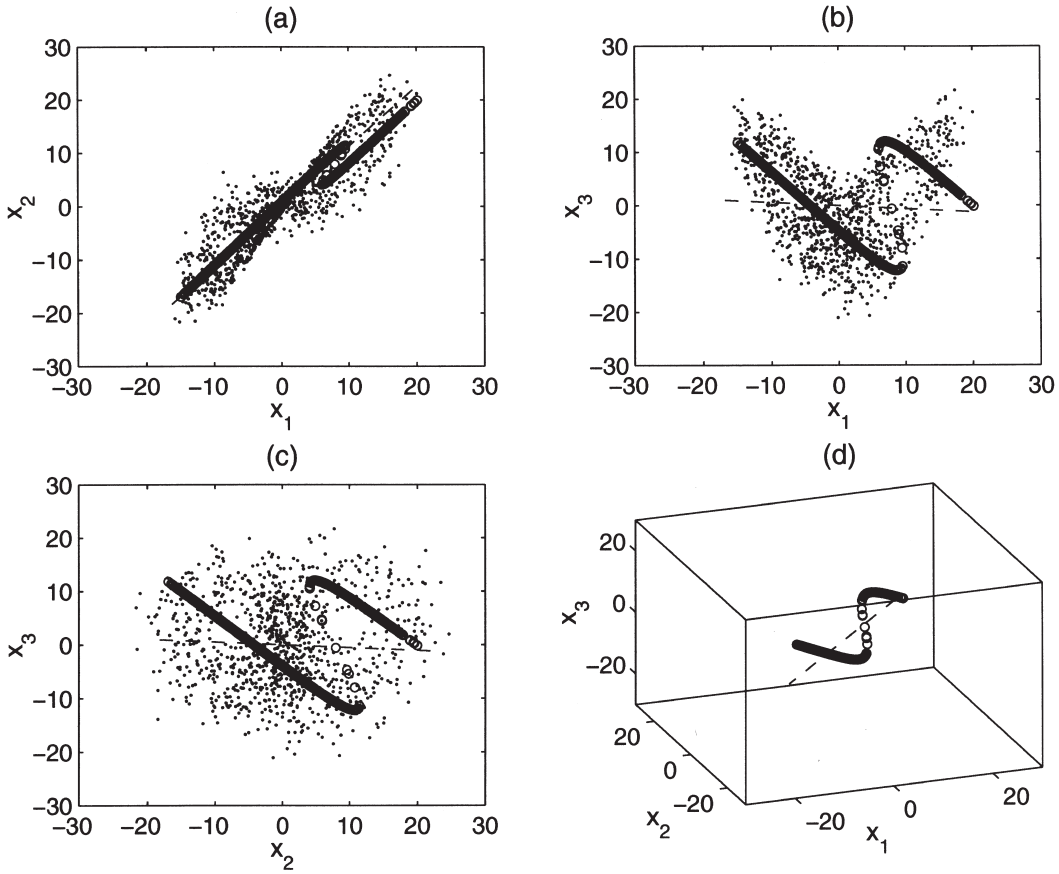

Fig. 2. The first NLPCA mode for data from the Lorenz (1963) system. The NLPCA mode is indicated by the (overlapping) circles, with the data shown as dots. Panel (a) displays the x_1–x_2 plane, (b) the x_1–x_3 plane and (c) the x_2–x_3 plane, and (d) gives a 3-D view. The butterfly-shaped attractor is most visible in panel (b). The NLPCA had $m = 2$, i.e., 2 hidden neurons in both the encoding and decoding layers. The dashed line shows the first PCA mode.

We have not penalized other weights in the network. In principle, $\boldsymbol{w}^{(u)}$ also controls the nonlinearity in the inverse mapping from u to $\boldsymbol{x}'$. However if the nonlinearity in the forward mapping from $\boldsymbol{x}$ to u is already limited, then there is no need to further limit the weights in the inverse mapping. The NLPCA is not very sensitive to the value of p; a value around 1 appears to work well here. The NLPCA with $p = 1$ applied to the Lorenz data yielded the U-shaped curve (Fig. 3), similar to that found by Monahan (2000).

Another example will shed more light on the advantage of using weight penalty terms. Let us generate two theoretical modes (Hsieh, 2000), with the first described by

$$X_1 = t - 0.3t^2, \qquad X_2 = t + 0.3t^3, \qquad X_3 = t^2, \tag{14}$$

where t is a random number uniformly distributed in the interval $[-1, 1]$. The 2nd theoretical mode is described by

$$X_1' = -s - 0.3s^2, \qquad X_2' = s - 0.3s^3, \qquad X_3' = -s^4, \tag{15}$$

with random number s uniformly distributed in $[-1, 1]$.

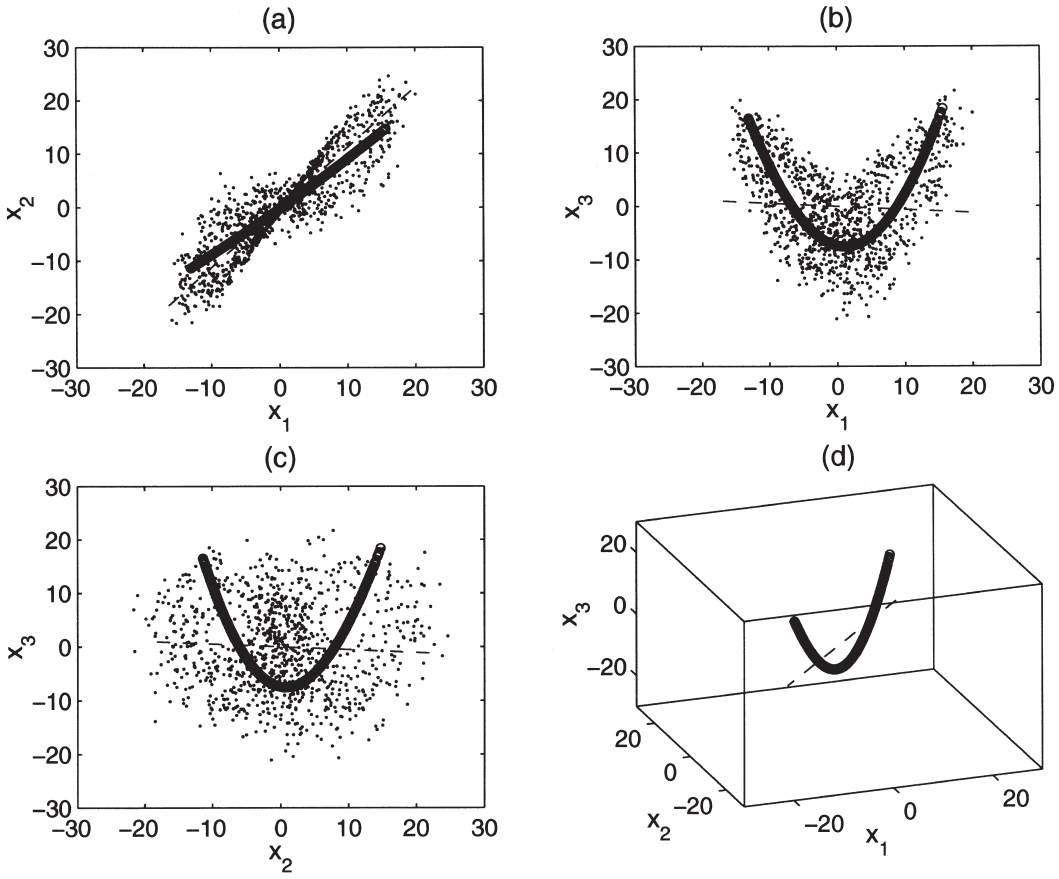

Fig. 3. The first NLPCA mode for the same Lorenz data, but with the weight penalty parameter $p = 1$. The NLPCA had $m = 2$.

To lowest order, $\boldsymbol{X}$ describes a quadratic curve and $\boldsymbol{X}'$ a quartic. A dataset of 500 points was generated by adding the second mode to the first mode, with the variance of the second mode being 1/3 that of the first mode. A small amount of Gaussian random noise, with standard deviation equal to 10% of the signal standard deviation, was also added to the dataset. The variables were then standardized (Fig. 4).

The NLPCA solution with $m = 2$ and no penalty ($p = 0$) extracted a solution resembling the first theoretical mode $\boldsymbol{X}$ (not shown). The MSE decreased from 0.667 for $m = 2$, to 0.636 for $m = 3$, and 0.599 for $m = 4$. At $m = 4$, the first NLPCA mode (Fig. 5) is a mixture of the 1st and 2nd theoretical modes. In contrast, if weight penalty is used ($p = 1$), then even with $m = 4$, the MSE is 0.717, and a solution resembling the theoretical first mode is found (Fig. 6). Thus weight penalty has prevented the mixing of the two theoretical modes.

If we then extract the second NLPCA mode from the residual left behind by the 1st NLPCA mode, the 2nd theoretical mode was successfully found for the $p = 1$ case, but not for the $p = 0$ case (not shown), as the first NLPCA had already mixed up the 1st and 2nd theoretical modes when $p = 0$. The MSE for the $p = 1$ case was only about 1/3 that for the $p = 0$ case (0.046 versus 0.140). The lesson is that even though a smaller MSE of the first NLPCA mode can be attained with $p = 0$ than with $p > 0$, it can be costly down the

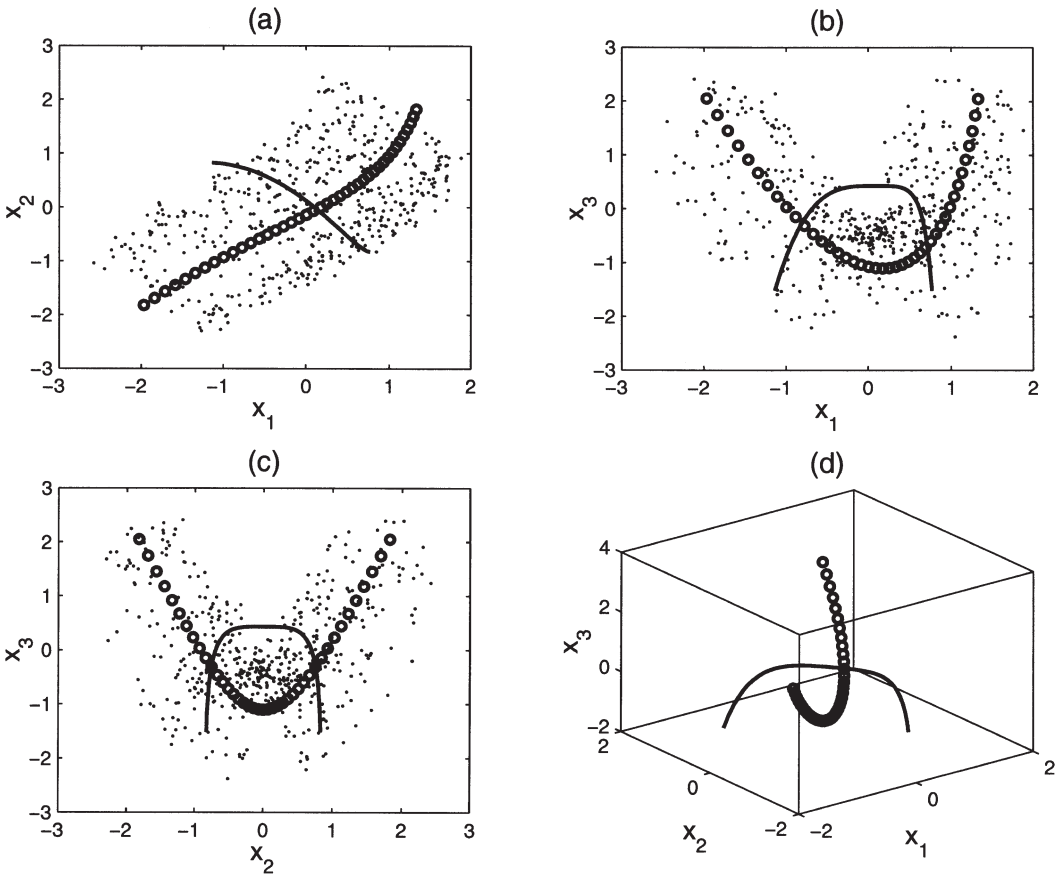

Fig. 4. The curve made up of circles shows the first theoretical mode (X), and the solid curve, the second theoretical mode (X'). The actual data set of 500 points (shown as dots) is generated by adding mode 2 to mode 1 (with mode 2 having 1/3 the variance of mode 1) and adding a small amount of Gaussian noise.

road, as the MSE of the second NLPCA mode for $p = 0$ may be much worse than that for $p > 0$.

4. A 3-way comparison between NLPCA, RPCA and PCA

In the linear approach, there is a dichotomy between PCA and RPCA. In PCA, the linear mode which accounts for the most variance of the dataset is sought. However, as illustrated in Preisendorfer (1988, Fig. 7.3), the resulting eigenvectors may not align close to local data clusters, so the eigenvectors may not represent actual physical states well. The RPCA methods rotate the PCA eigenvectors, so they point closer to the local clusters of data points. Thus the rotated eigenvectors may bear greater resemblance to actual physical states (though they account for less variance) than the unrotated eigenvectors, so RPCA is also widely used (Barnston and Livezey, 1987). As there are many possible criteria for rotation, there are many RPCA schemes, among which the varimax (Kaiser, 1958) scheme is perhaps the most popular.

In this section, we use the tropical Pacific SST to make a 3-way comparison between NLPCA, RPCA and PCA. The monthly Pacific SST data from NOAA (Reynolds and Smith, 1994; Smith et al., 1996) for the period January, 1950 to April, 1999 were used. The 2° by 2° resolution data, covering the region 30°S to 30°N and 120°E to

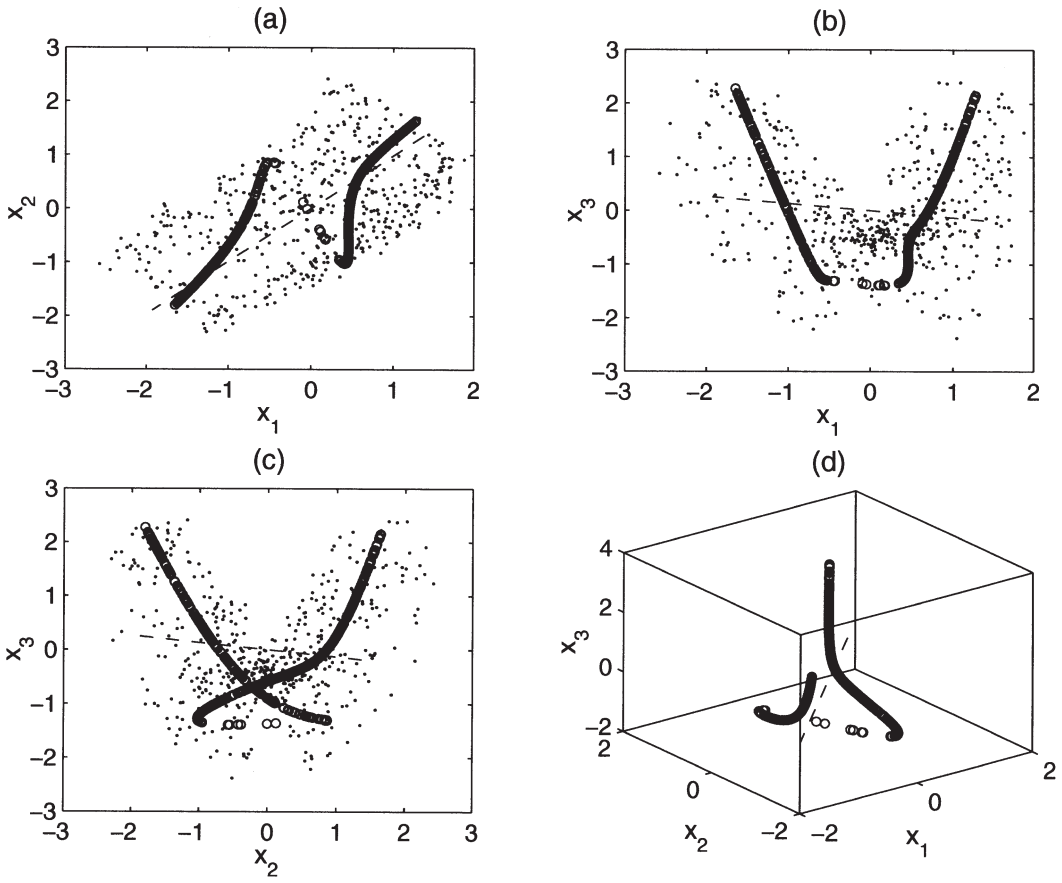

Fig. 5. The first NLPCA mode extracted with $m = 4$ and $p = 0$ (no weight penalty), showing a mixing of the first and second theoretical modes from Fig. 4. The dash line shows the first PCA mode.

60°W, were combined into 4° by 4° gridded data and smoothed by a 3-month running average. Thus there are $(15 \times 45 =)$ 675 spatial variables and 592 time points. There are still far too many spatial variables for this dataset to be directly analyzed by the NLPCA. With $l = 675$, the smallest nonlinear NLPCA model (with $m = 2$) would contain $(2lm + 4m + l =)$ 3383 parameters, which greatly exceed the number of time points.

To reduce the number of input variables, prefiltering the SST data by PCA is needed. PCA modes 1, 2 and 3 (Fig. 7) accounted for 51.4%, 10.1% and 7.2%, respectively, of the variance in the SST data. The equatorial Pacific is known for its warm states (El Niño) and cool states (La Niña), which are manifested in the first mode. The second mode represents the asymmetry between El Niño and La Niña, as cool anomalies associated with La Niña events are centred further west of the warm anomalies of El Niño (Hoerling et al., 1997). The first 3 PCs (PC1, PC2 and PC3) were used as the input $\boldsymbol{x}$ for the NLPCA network — the number of inputs is kept small for pedagogical reasons.

The data are shown as dots in a scatter plot in the PC1–PC2 plane (Fig. 8), where the cool La Niña states lie in the upper left corner, and the warm El Niño states in the upper right corner. The NLPCA (with $m = 2$) solution is a U-shaped curve linking the La Niña states at one end (low u) to the El Niño states at the other end (high u), similar to that found by Monahan (2001). In contrast, the first PCA eigenvector lies along the horizontal line, and the second PCA, along the

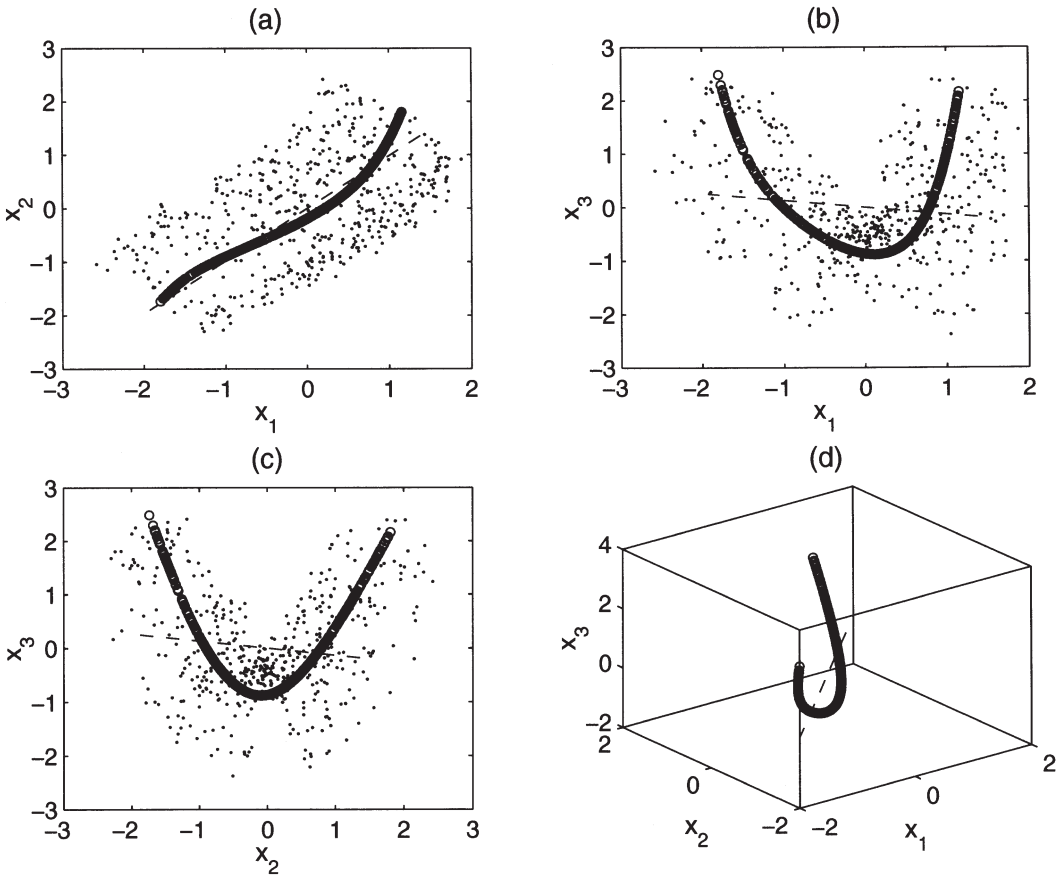

Fig. 6. The first NLPCA mode extracted with $m = 4$ and $p = 1$, showing a good resemblance to the first theoretical mode $\boldsymbol{X}$.

vertical line (Fig. 8), neither of which would come close to the El Niño nor the La Niña states.

With the NLPCA, for a given value of u, one can map from u to the 3 PCs. Each of the 3 PCs can be multiplied by its associated PCA (spatial) eigenvector, and the three added together to yield the spatial pattern for that particular value of u. Figs 9a–d show the spatial anomaly patterns as u goes from its minimum value (corresponding to the strongest La Niña), all the way to its maximum value (corresponding to the strongest El Niño). Clearly the asymmetry between El Niño and La Niña is well captured by the first NLPCA mode, as found by Monahan (2001). Incidentally, Monahan (2001) used the first 10 PCs as inputs to his NLPCA, versus only 3 here. Since the results are very similar, this means the essense of the El Niño/La Niña mode is contained within the first 3 PCA modes.

In Fig. 8, the first PCA eigenvector spears neither the cluster of El Niño states in the upper right corner nor the La Niña states in the upper left corner, and is therefore not particularly good in representing either. For comparison, a varimax rotation (Kaiser 1958; Preisendorfer 1988), was applied to the first 3 PCA eigenvectors. The resulting first RPCA eigenvector, shown as a dashed line in Fig. 8, spears through the cluster of El Niño states in the upper right corner, thereby yielding a more accurate description of the El Niño anomalies (Fig. 9e) than the first PCA mode (Fig. 7a), which did not fully represent the intense warming of Peruvian waters. The second RPCA eigenvector, also shown as a dashed line in Fig. 8,

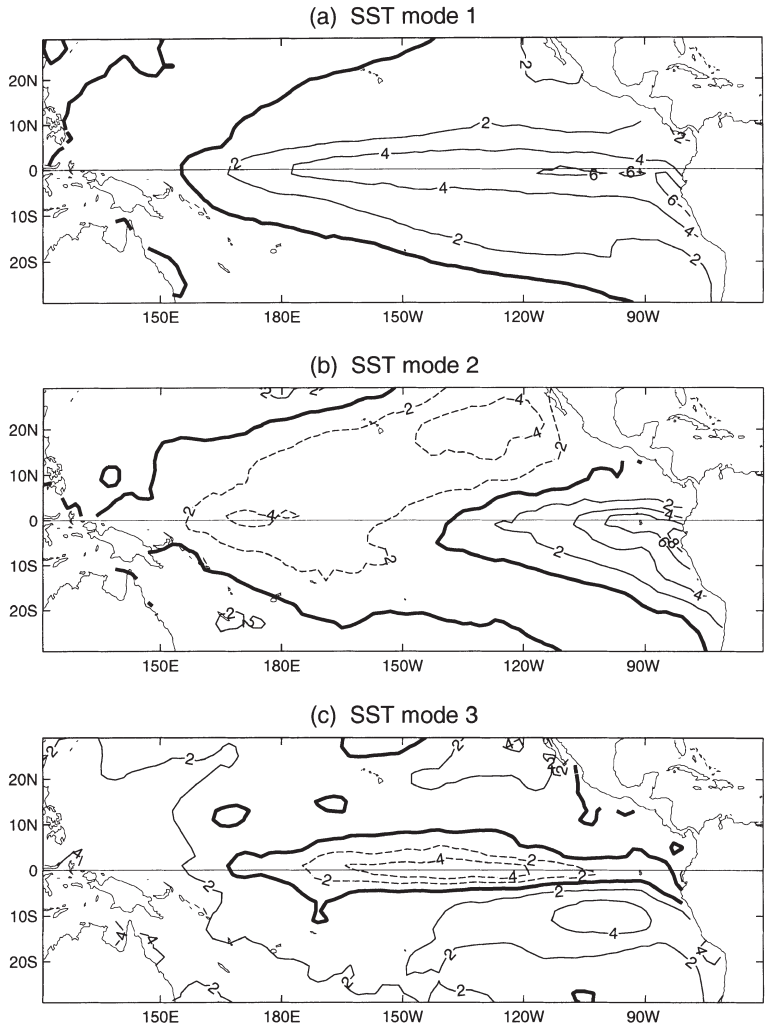

Fig. 7. The first 3 PCA spatial modes (i.e., eigenvectors) of the tropical Pacific monthly SST (where the climatological seasonal cycle had been removed). The eigenvectors have been normalized to unit norm, and the contours are in units of 0.01°C, with the positive contours shown as solid curves, negative contours, dashed curves, and the zero contour, a thick curve.

did not improve much on the second PCA mode, with the RPCA spatial pattern shown in Fig. 9f (cf Fig. 7b).

One problem with RPCA is that there are many possible ways to rotate the PCA eigenvectors. In fact, nineteen types of rotations are listed in Richman (1986), rendering a certain amount of arbitrariness to RPCA when compared with PCA. In our example here, while the El Niño states are well represented by a RPCA eigenvector, the La Niña states are not represented by a single RPCA eigenvector. In contrast, the first NLPCA mode successfully passes through the La Niña states and the El Niño states as u varies continuously from its minimum value to its maximum value. In terms of variance explained, the first NLPCA mode explained 56.6% of the variance, versus 51.4% by the first PCA mode, and 47.2% by the first RPCA mode.

Here both RPCA and NLPCA take the PCs

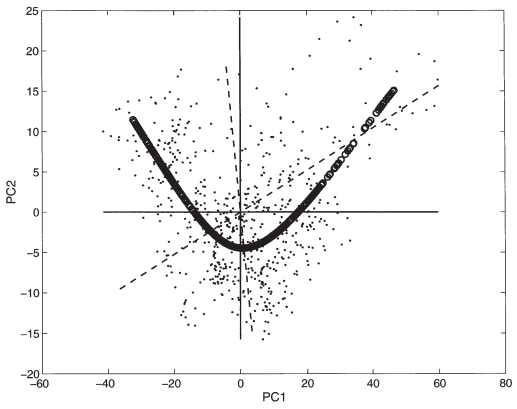

Fig. 8. Scatter plot of the SST data (shown as dots) in the PC1–PC2 plane, with the El Niño states lying in the upper right corner, and the La Niña states in the upper left corner. The PC2 axis is stretched relative to the PC1 axis for better visualization. The first mode NLPCA approximation to the data is shown by the small circles, which traced out a *U*-shaped curve. The first PCA eigenvector is indicated by the horizontal line, and the second PCA, by the vertical line. The varimax method rotates the two PCA eigenvectors in a counterclockwise direction, as the rotated PCA (RPCA) eigenvectors are indicated by the dashed lines. (As the varimax method generates an orthogonal rotation, the angle between the two RPCA eigenvectors is 90° in the 3-dimensional PC1–PC2–PC3 space).

from PCA as input. However, instead of multiplying the PCs by a fixed orthonormal rotational matrix, as performed in the varimax RPCA approach, NLPCA performs a nonlinear mapping of the PCs. It is easy to see why the dichotomy between PCA and RPCA in the linear approach automatically vanishes in the nonlinear approach. By increasing m to a large enough value in NLPCA, the solution is capable of going through all local data clusters while maximizing the global variance explained. (In fact, for large enough m, NLPCA can pass through all data points, though this will in general give an undesirable, overfitted solution.)

5. NLPCA with a circular bottleneck node

The NLPCA of Kramer (1991) is capable of extracting open curve solutions, but not closed curve solutions, as the bottleneck neuron u is not an angular variable. Kirby and Miranda (1996) introduced a circular node or neuron, and showed that the NLPCA with a circular node at the bottleneck is capable of extracting closed curve solutions. Fig. 10 shows the circular-noded NLPCA (henceforth abbreviated as NLPCA.cir), which is identical to the NLPCA of Fig. 1, except at the bottleneck, where there are now two coupled neurons p and q, instead of a single neuron u.

Analogous to u in (7), we calculate the prestates p_0 and q_0 by

$$p_0 = \boldsymbol{w}^{(x)} \cdot \boldsymbol{h}^{(x)} + \bar{b}^{(x)}, \qquad q_0 = \tilde{\boldsymbol{w}}^{(x)} \cdot \boldsymbol{h}^{(x)} + \tilde{b}^{(x)}, \tag{16}$$

where $\boldsymbol{w}^{(x)}$, $\tilde{\boldsymbol{w}}^{(x)}$ are weight parameter vectors, and $\bar{b}^{(x)}$ and $\tilde{b}^{(x)}$ are bias parameters. Let

$$r = (p_0^2 + q_0^2)^{1/2}, \tag{17}$$

then the circular node is defined with

$$p = p_0/r, \qquad q = q_0/r, \tag{18}$$

satisfying the unit circle equation $p^2 + q^2 = 1$. Thus, even though there are two variables p and q at the bottleneck, there is only one angular degree of freedom from θ (Fig. 10), due to the circle constraint. The mapping from the bottleneck to the output proceeds as in Section 2, with (5) replaced by

$$h_k^{(u)} = \tanh[(\boldsymbol{w}^{(u)} p + \tilde{\boldsymbol{w}}^{(u)} q + \boldsymbol{b}^{(u)})_k]. \tag{19}$$

When implementing NLPCA.cir, I found that there are actually two possible configurations: (i) a restricted configuration where the constraint $\langle p \rangle = \langle q \rangle = 0$ is applied, and (ii) a general configuration where no constraint on $\langle p \rangle$ and $\langle q \rangle$ is applied. With (i), there are two fewer free parameters, as

$$\bar{b}^{(x)} = -\langle \boldsymbol{w}^{(x)} \cdot \boldsymbol{h}^{(x)} \rangle, \qquad \tilde{b}^{(x)} = -\langle \tilde{\boldsymbol{w}}^{(x)} \cdot \boldsymbol{h}^{(x)} \rangle. \tag{20}$$

If a closed curve solution is sought, then (i) is better than (ii) as it has two fewer parameters. However, (ii), being more general than (i), can actually model open curve solutions like a regular NLPCA. The reason is that if the input data mapped onto the p–q plane covers only a segment of the unit circle instead of the whole circle, then the inverse mapping from the p–q space to the output space will yield a solution resembling an open curve. Hence, given a dataset, (ii) may yield either a closed curve or an open curve solution. Its generality comes with a price, namely that

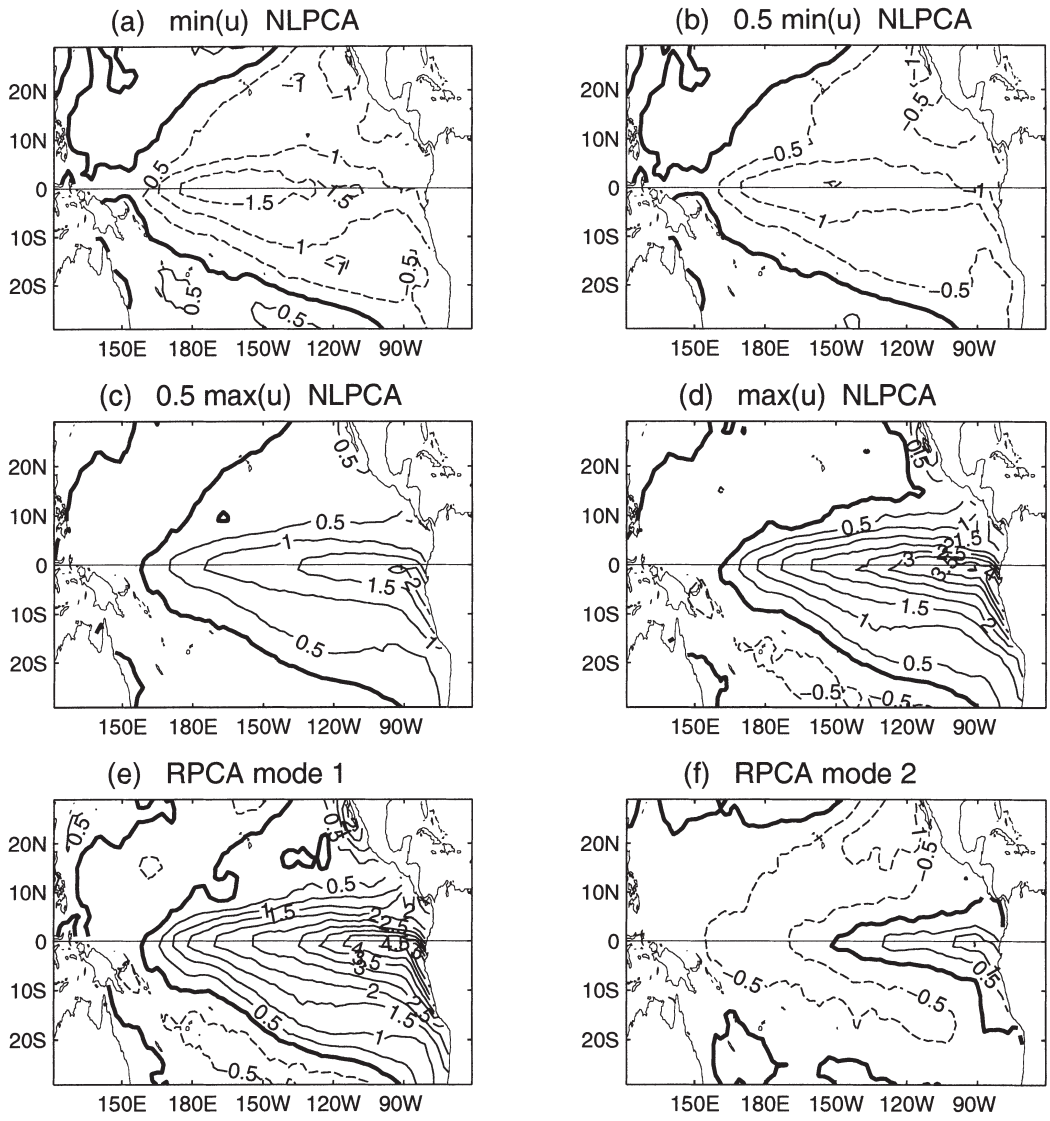

Fig. 9. The SST anomaly patterns (°C) of the NLPCA and the RPCA. The anomaly pattern as the NLPC u of the first NLPCA mode varies from (a) its minimum (strong La Niña), to (b) half its minimum (weak La Niña), to (c) half its maximum (weak El Niño) and (d) its maximum (strong El Niño). The first and second varimax RPCA spatial modes are shown in (e) and (f) respectively, (both with their corresponding RPCs at maximum value). With a contour interval of 0.5°C, the positive contours are shown as solid curves, negative contours, dashed curves, and the zero contour, a thick curve. When comparing these patterns with those in Fig. 7, remember that the patterns in Fig. 7 are normalized differently (i.e., with unit norm).

there may be more local minima to contend with. The number of free parameters is $2lm + 6m + l$ for configuration (i), and $2lm + 6m + l + 2$ for (ii). Unlike NLPCA which reduces to PCA when only linear transfer functions are used, NLPCA.cir does not appear to have a linear counterpart.

Next we apply NLPCA.cir to the tropical Pacific SST. The data set is the same as that used

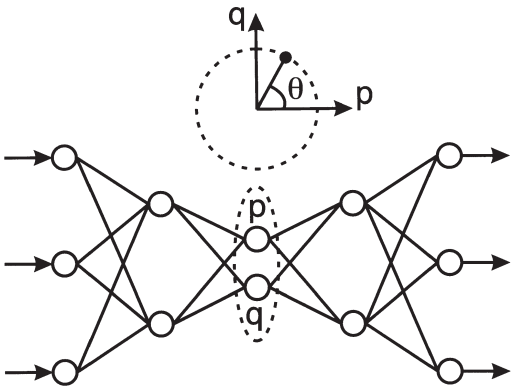

Fig. 10. The NLPCA with a circular node at the bottleneck. Instead of having one bottleneck neuron u as in Fig. 1, there are now two neurons p and q constrained to lie on a unit circle in the p–q plane, so there is only one free angular parameter (θ).

in Section 4, but the climatological seasonal cycle has not been removed from the data. PCA was then performed, with mode 1, 2 and 3 accounting for 78.8%, 10.8% and 3.4%, respectively, of the total variance. The PC1 time series of the first mode is now totally dominated by the seasonal cycle, and the spatial pattern (not shown) displays the SST anomalies associated with the Northern Hemisphere winter, i.e., a spatial pattern dominated by the meridional temperature gradient, with cooler waters to the north and warmer waters to the south. The second mode, which has a spatial pattern resembling that of Fig. 7a, has its PC time series showing El Niño and La Niña events, as well as the seasonal cycle. The mode 3 PC time series shows the seasonal cycle and interdecadal fluctuations. These three PC time series are then input into NLPCA.cir.

Since we want to extract the dominant seasonal cycle, we use the restricted configuration for NLPCA.cir (i.e., with $\langle p \rangle = \langle q \rangle = 0$). The first mode extracted the seasonal cycle (Fig. 11), which illustrates how NLPCA.cir can extract periodic or wave modes from the data. Of course, this is a trivial example as the period of the seasonal cycle is well defined, so it can easily be extracted without using NLPCA.cir. But for waves of unknown period or quasi-periodic waves, NLPCA.cir may be valuable in their extraction — the retrieved curve can be a continuous closed curve of any shape (if an adequate number of hidden neurons is used in the encoding and decoding layers).

From the residual, we extracted the second mode (Fig. 12) — using the general configuration (no constraint on $\langle p \rangle$ and $\langle q \rangle$), as we are not sure a priori whether to expect an open curve or a closed curve. The second mode resulted in an open curve (though some of the ensemble members ended up at shallower local minima corresponding to closed curves). In Fig. 12a, the El Niño states are located at the upper right corner, and the La Niña states at the lower right corner. That high PC2 values (El Niño conditions) tend to occur with high PC1 (which occurs during winter) simply means that the peak of El Niño events tend to occur during winter. That low PC2 (La Niña events) also tend to occur during moderately high values of PC1 means that the La Niña events also tend to peak during winter.

For NLPCA.cir mode 2, the p, q values cover only a segment of the circle, with θ ranging from -182° to 67°. The SST anomalies associated with maximum θ (Fig. 13a) correspond to La Niña. Comparing with the La Niña picture of Fig. 9a, we see that Fig. 13a gives the additional information that La Niña tends to occur during winter, as SST north of the equator are cooler than SST south of the equator. Fig. 13b shows the minimum θ situation. Here El Niño warming of the eastern equatorial water occurs when SST north of the equatorial region are cooler than SST south of the equatorial region, i.e., during winter. Thus NLPCA.cir mode 2 shows the El Niño and La Niña states in their proper relation with the seasonal cycle. Incidentally, the seasonal cycle is almost invisible in the second mode θ time series (not shown), in contrast to PC2 (not shown), where the seasonal cycle is manifested. This means that the first NLPCA.cir mode has fully captured the seasonal cycle, unlike PCA mode 1, which only captured part of the seasonal cycle, scattering the remainder into PC2 and PC3.

If NLPCA.cir (general configuration) is applied to the data in Section 4 (i.e., with climatological seasonal cycle removed prior to the PCA), the resulting first mode is indistinguishable from that obtained by the original NLPCA (Figs. 8, 9a,b,c,d).

6. Conclusions

The advent of neural network (NN) models has significantly advanced nonlinear empirical model-

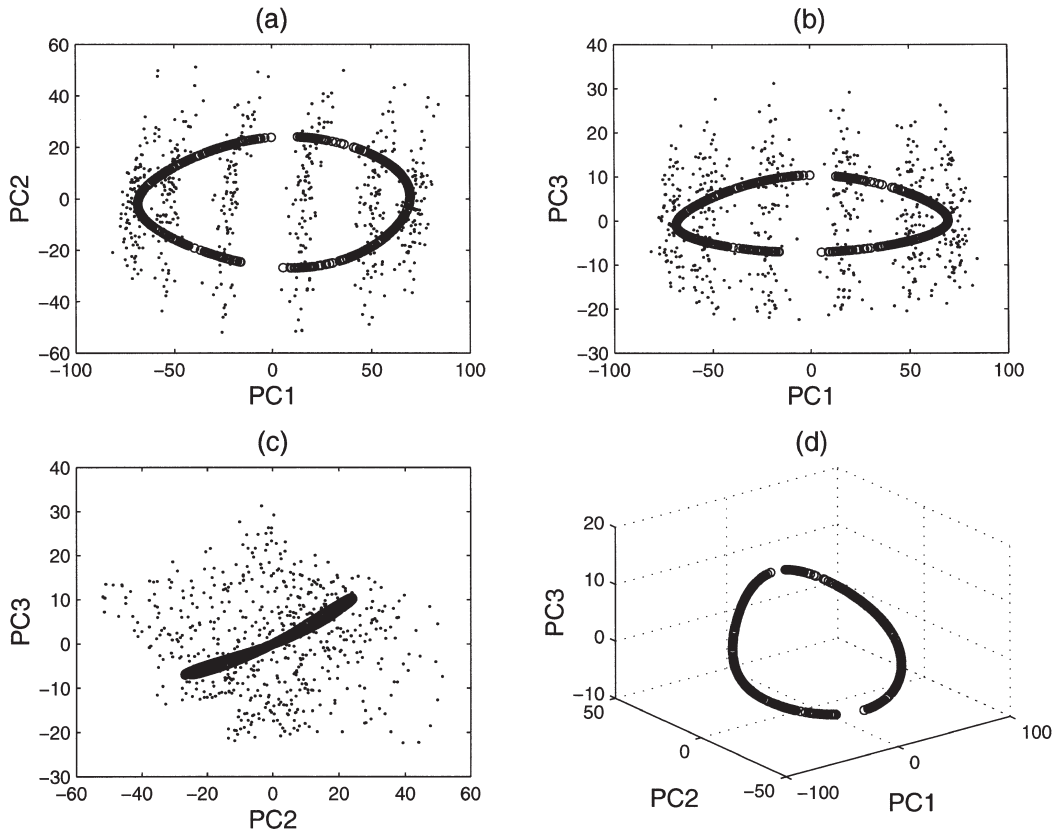

Fig. 11. The first NLPCA.cir (NLPCA with circular bottleneck node) mode extracted with $m = 3$ and $p = 1$, shown as (overlapping) circles, with the data as dots.

ling. Every member of the hierachy of classical multivariate methods — multiple linear regression, PCA and canonical correlation analysis (CCA) — has been nonlinearly generalized by NN models — nonlinear multiple regression by Rumelhart et al. (1986), nonlinear PCA by Kramer(1991), and nonlinear CCA by Hsieh (2000, 2001). The nonlinear PCA and CCA codes are available from the author's web site, http://www.ocgy.ubc.ca/projects/clim.pred.

PCA is used for two main purposes: (i) to reduce the dimensionality of the dataset, and (ii) to extract features or recognize patterns from the dataset. It is purpose (ii) where PCA can be improved upon. Rotated PCA (RPCA) sacrifices on the amount of variance explained, but by rotating the PCA eigenvectors, RPCA eigenvectors tend to point more towards local data clusters and are therefore more representative of physical states than the PCA eigenvectors. With the tropical Pacific SST as an example, it was shown that RPCA represented El Niño states better than PCA, but neither methods represented La Niña states well. In contrast, nonlinear PCA (NLPCA), passed through both the clusters of El Niño and La Niña states, thus representing both well within a single mode. Furthermore, the NLPCA first mode explained more variance of the dataset than the first mode of PCA or RPCA. With a linear approach, it is generally impossible to have a solution simultaneously (a) explaining maximum global variance of the dataset and (b) approaching local data clusters, hence the dichotomy between PCA and RPCA. With the more flexible NLPCA

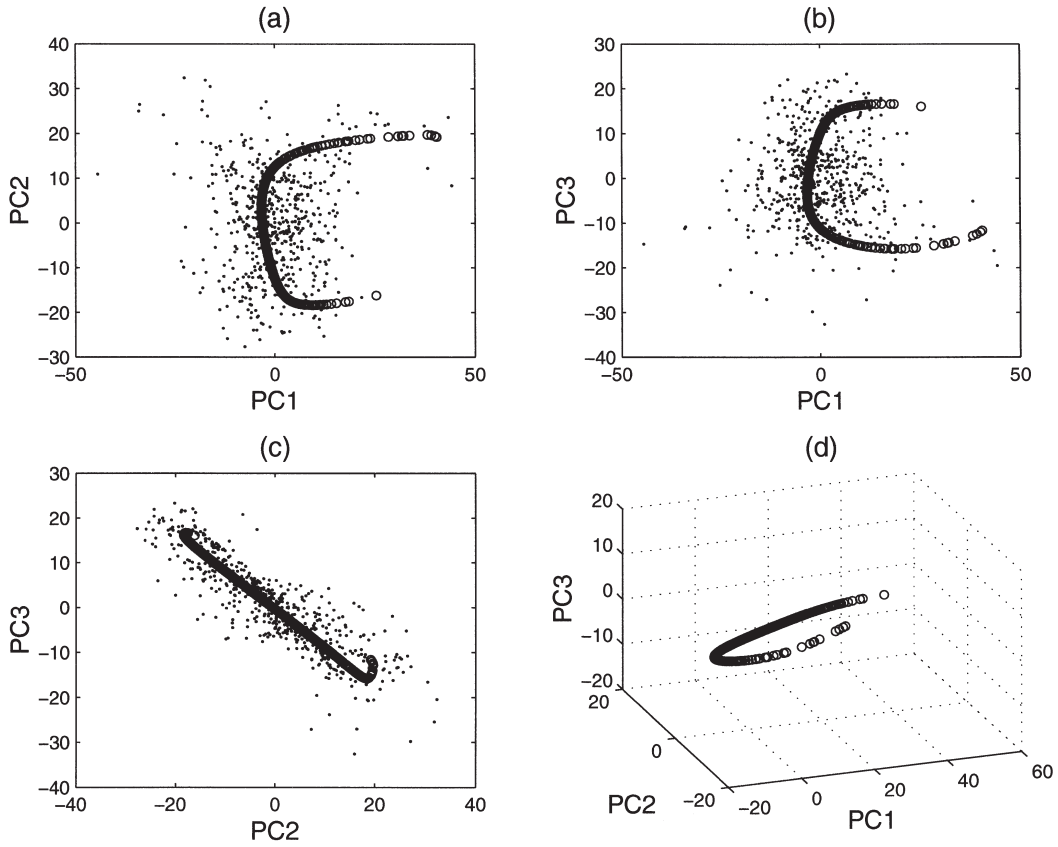

Fig. 12. The second NLPCA.cir mode extracted with $m = 3$ and $p = 1$, shown as circles, with the data as dots.

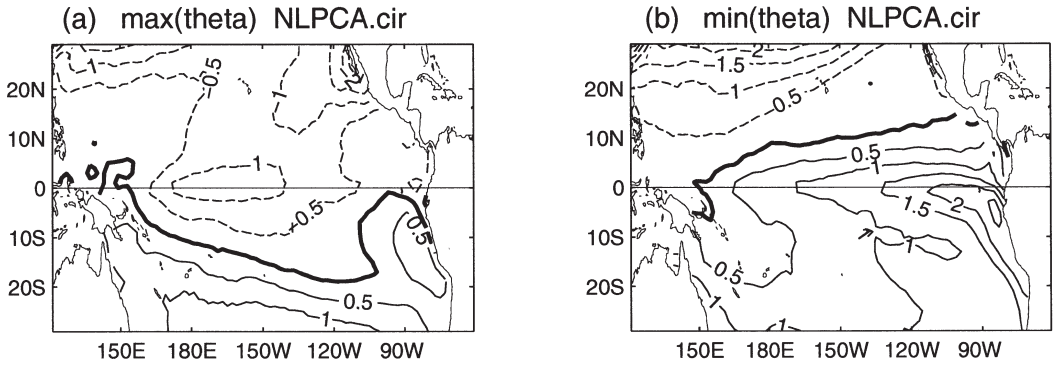

Fig. 13. The SST anomaly patterns (°C) of the second NLPCA.cir mode when (a) θ is maximum (strong La Niña), and (b) θ is minimum (strong El Niño).

method, both objectives (a) and (b) can be attained together, thus the nonlinearity in NLPCA unifies the PCA and RPCA approaches.

The main disadvantage of NLPCA compared with the linear methods lies in its instability or nonuniqueness — with multiple minima in the cost function, optimizations started from different initial parameters often end up at different minima for the NLPCA. An ensemble of optimization runs starting from different random initial parameters is needed, where the best ensemble member is chosen as the solution — even then, there is no guarantee that the global minimum has been found. Proper scaling of the input data is essential to avoid having the nonlinear optimization algorithm searching for parameters with a wide range of magnitudes. Regularization by adding weight penalty terms to the cost function greatly improved the stability of the NLPCA. Only the weights in the encoding layer of the network need to be penalized to avoid excessive nonlinearity in the solution. Weight penalty was found to be effective in preventing zigzag shaped solutions and the mixing of two theoretical modes.

With PCA, the straight line explaining the maximum variance of the data is found. With NLPCA, the straight line is replaced by a continuous, open curve. NLPCA.cir (NLPCA with a circular node at the bottleneck) replaces the open curve with a closed curve, so periodic or wave solutions can be modelled. When dealing with data containing a nonlinear or periodic structure, the linear methods scatter the energy into multiple modes, which is usually prevented when the nonlinear methods are used.

Whether the nonlinear approach has a significant advantage over the linear approach is highly dependent on the dataset — the nonlinear approach is generally ineffective if the data record is short and noisy, or the underlying physics is essentially linear. Presently, m, the number of hidden neurons in the encoding layer, and p, the weight penalty parameter, are determined largely by a trial and error approach. Future research will hopefully provide more guidance on their choice.

7. Acknowlegments

Benyang Tang kindly sent me the SST dataset and his Matlab contouring package. Helpful comments were provided by members of our research group, especially Adam Monahan, Youmin Tang, Aiming Wu and Yuval. Support through research and strategic grants from the Natural Sciences and Engineering Research Council of Canada is gratefully acknowledged.

REFERENCES

Barnston, A. G. and Livezey, R. E. 1987. Classification, seasonality and persistence of low-frequency atmospheric circulation patterns. *Mon. Wea. Rev.* **115**, 1083–1126.

Cybenko, G. 1989. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems* **2**, 303–314.

Diamantaras, K. I. and Kung, S. Y. 1996. *Principal component neural networks*. New York: Wiley.

Hoerling, M. P., Kumar, A. and Zhong, M. 1997. El Niño, La Niña and the nonlinearity of their teleconnections. *J. Climate* **10**, 1769–1786.

Hsieh, W. W. 2000. Nonlinear canonical correlation analysis by neural networks. *Neural Networks* **13**, 1095–1105.

Hsieh, W. W. 2001. Nonlinear canonical correlation analysis of the tropical Pacific climate variability using a neural network approach. *J. Climate*, in press.

Hsieh, W. W. and Tang, B. 1998. Applying neural network models to prediction and data analysis in meteorology and oceanography. *Bull. Amer. Meteor. Soc.* **79**, 1855–1870.

Jollife, I. T. 1986. *Principal component analysis*. New York: Springer.

Kaiser, H. F. 1958. The varimax criterion for analytic rotation in factor analysis. *Psychometrika* **23**, 187–200.

Kirby, M. J. and Miranda, R. 1996. Circular nodes in neural networks. *Neural Comp.* **8**, 390–402.

Kramer, M. A. 1991. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal* **37**, 233–243.

Lorenz, E. N. 1963. Deterministic nonperiodic flow. *J. Atmos. Sci.* **20**, 130–141.

Monahan, A. H. 2000. Nonlinear principal component analysis by neural networks: theory and application to the Lorenz system. *J. Climate* **13**, 821–835.

Monahan, A. H. 2001. Nonlinear principal component analysis: tropical Indo-Pacific sea surface temperature and sea level pressure. *J. Climate* **14**, 219–233.

Oja, E. 1982. A simplified neuron model as a principal component analyzer. *J. Math. Biology* **15**, 267–273.

Preisendorfer, R. W. 1988. *Principal component analysis in meteorology and oceanography*. New York: Elsevier.

Reynolds, R. W. and Smith, T. M. 1994. Improved global sea surface temperature analyses using optimum interpolation. *J Climate* **7**, 929–948.

Richman, M. B. 1986. Rotation of principal components. *J. Climatology* **6**, 293–335.

Rumelhart, D. E., Hinton, G. E. and Williams, R. J. 1986. Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, and P. R. Group (Eds.), *Parallel distributed processing* (vol. 1, pp. 318–362). Cambridge, MA: MIT Press.

Smith, T. M., Reynolds, R. W., Livezey, R. E. and Stokes, D. C. 1996. Reconstruction of historical sea surface temperatures using empirical orthogonal functions. *J. Climate* **9**, 1403–1420.

Von Storch, H. and Zwiers, F. W. 1999. *Statistical analysis in climate research*. Cambridge: Cambridge Univ. Pr.