Available online at www.sciencedirect.com
SCIENCE DIRECT

ELSEVIER

Neurocomputing 57 (2004) 37–47

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Forecasting using twinned principal curves and twinned self-organising maps

Ying Han, Emilio Corchado, Colin Fyfe*

Applied Computational Intelligence Research Unit, The University of Paisley, Paisley, Scotland, UK

Abstract

We extend the principal curves algorithm by creating twinned principal curves which extend through two related data sets simultaneously. The criteria for accepting a pair of data points as neighbours for any other pair of data points is that each of the relevant points must be close in the appropriate space. We illustrate the algorithm's predictive power on artificial data sets before using it to predict on a real financial time series. We compare the error from this twinning with that achieved by a related algorithm which twins self-organising maps.

Keywords: Principal curves; Twinned mappings

1. Introduction

This paper discusses the effect of twinning two data sets and applying algorithms which are normally only applied to a single data set to the pair of twinned sets. We will twin the principal curves algorithm [6] and Kohonen's self-organising map (SOM) [8] and use the resulting algorithms to forecast time series data: one data set is the previous values of the time series while the second data set is the future values of the time series. Both methods give nonlinear projections which suits the task of forecasting on typical data sets in which there is no linear relationship between the past and the future. Both methods can be thought of as extensions of linear techniques for relating two data sets.

The statistical technique for estimating the linear combination of a data set which gives the greatest correlation with a linear combination of a second data set is known as canonical correlation analysis (CCA). Let $\mathbf{x}_1$ be a vector drawn from the first data set and let $\mathbf{x}_2$ be the corresponding vector drawn from the second data set. Then CCA

* Corresponding author. Fax: +44-141-848-3305.
E-mail address: colin.fyfe@paisley.ac.uk (C. Fyfe).

0925-2312/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2004.01.009

attempts to estimate $\mathbf{w}_1$ and $\mathbf{w}_2$ such that $y_1 = \mathbf{w}_1^T\mathbf{x}_1$ and $y_2 = \mathbf{w}_2^T\mathbf{x}_2$ have the greatest correlation over the whole set of samples $\mathbf{x}_1$ and $\mathbf{x}_2$. We have previously developed neural algorithms [3,9,11] for performing CCA; the neural algorithms have certain advantages over standard statistical techniques including the ability to find nonlinear projections of a data set which maximise correlations. We have also used the neural algorithms for forecasting [10]: one data set is the previous samples of a time series, the other is the sample(s) which one wishes to predict.

We have previously [5] developed an extension of principal curves which performs a type of nonparametric CCA. We illustrate its use on artificial data and then use the method to forecast on a financial data set which we have previously [4] used to test other forecasting methods. We then develop a twinned SOM algorithm and compare its ability to forecast with that of the twinned principal curves.

2. Twinned principal curves

Principal component analysis (PCA) is a standard statistical technique for finding a lower dimensional linear projection of high dimensional data which gives minimum mean square error over all projections of this dimensionality. Principal curves [1,6,7] is an extension of this method in which a nonlinear manifold can be used instead of the linear subspace determined by PCA. However there is clearly a difficulty with this in that it is always possible to fit a finite training set with no error. There are several definitions of principal curves which constrain the curves in one way or another to overcome the problem of overfitting. In [6], every point, P, on the curve is the mean of the points that project onto P. This is known as self-consistency. The unit-speed curve (one whose derivative has norm 1) which satisfies this is the principal curve. In [7], the principal curve is defined as the curve of a specific length which minimises the mean squared distance from the data.

In this paper, we extend the principal curve method so that we now find a nonlinear manifold in each of two data sets. We use a nonparametric method to determine the two manifolds. Since we are drawing data iid from two data sets simultaneously, our method creates manifolds which exhibit a correlation between corresponding points on the manifolds which we can then use to subsequently forecast a sample from one data set given a sample from the other. The algorithm in outline is

(1) Initialise d_1^i with the projection of $\mathbf{x}_1^i$ onto the first principal component of the first data set and similarly with d_2^i, $\forall i$.
(2) With the current projections d_1^i and d_2^i, $\forall i$.
(3) Select $\mathbf{x}_1^i$ from the first data set and the corresponding point, $\mathbf{x}_2^i$ from the second data set.
(4) Find all neighbours of the point which have
    - projections close to the projections of the chosen point.
    - projections of their corresponding points in the other data set satisfying the same constraint with respect to the second data set. Note that these projections will be to different curves.

(5) Thus, if d_1^i is the projection of $\mathbf{x}_1^i$ and d_2^i is the projection of $\mathbf{x}_2^i$, then $S_i = \{k : |d_1^k - d_1^i| < \varepsilon_1 \text{ and } |d_2^k - d_2^i| < \varepsilon_2\}$.
(6) Find the local average of points projecting close to $\mathbf{x}_1^i$ and $\mathbf{x}_2^i$. i.e. d_1^i (new) = Mean of $d_1^j, j \in S_i$ and d_2^i (new) = Mean of $d_2^j, j \in S_i$.
(7) $d_1^i = d_1^i$ (new) and $d_2^i = d_2^i$ (new), $\forall i$.
(8) Return to Step 2.

The algorithm iterates till a stopping criteria is met: either the algorithm repeats for a set number of rounds or till the number of nodes to which the data is projected reaches a certain number (see below) or till the mean square error reaches a particular value. We will in the following call the nodes to which the data project the "knot points" of the algorithm.

Clearly, there are extensions which can be made to this algorithm. For example it is possible to change the value of the width parameters ε_1 and ε_2 during the course of the iterations, though this is not implemented in the simulations discussed in this paper for reasons which will become clear in the next section. Also, the use of a weighted average rather than a simple average may improve the accuracy of the new projections. Finally, the algorithm tends to draw data from the extremes of the principal curve and so some additional local averaging may be useful in this case. Again the last two points are not implemented in the results discussed in this paper.

3. Experiments

3.1. Artificial data

We first create 2 sets of two-dimensional artificial data which are known to have a correlation from $x_1(t) = \sin(t) + \mu_1$, $y_1(t) = \cos(t) + \mu_2, x_2(t) = t + \mu_3$, $y_2(t) = (t/3) + \sin(t) + \mu_4$, where t is drawn from a uniform distribution in $[0, 2\pi]$ and $\mu_i - N(0, 0.2)$ is Gaussian noise. Examples of this data are shown in the top row of Fig. 1.

Fig. 1 also shows the thinning which takes place in data set 2 after 1, 2 and 10 iterations and in data set 1 after 10 iterations. The sparsification discussed above is clearly evident.

Now we may use these projections to predict the position of a point, $\mathbf{x}_2$, in data set 2 given its corresponding point $\mathbf{x}_1$ in data set 1. Typically, we will approximate the principal curves with the sum of linear projections given by joining the sparse points as shown in the last row of Fig. 1. To forecast, we project $\mathbf{x}_1$ onto the current principal curve of the first data set and use the corresponding point on the current principal curve of the second data as the predictor of $\mathbf{x}_2$. Typical results are shown in the last row of Fig. 1, the "$*$" on the curve being the predictor while the "$+$" shows the point's actual position.

3.2. Forecasting

The problem we have modelled is a forecasting one: given the last few days' exchange rates (US dollar against Japanese yen), is it possible to forecast the next day's exchange rate with some degree of accuracy? We have previously [4] used a variety of

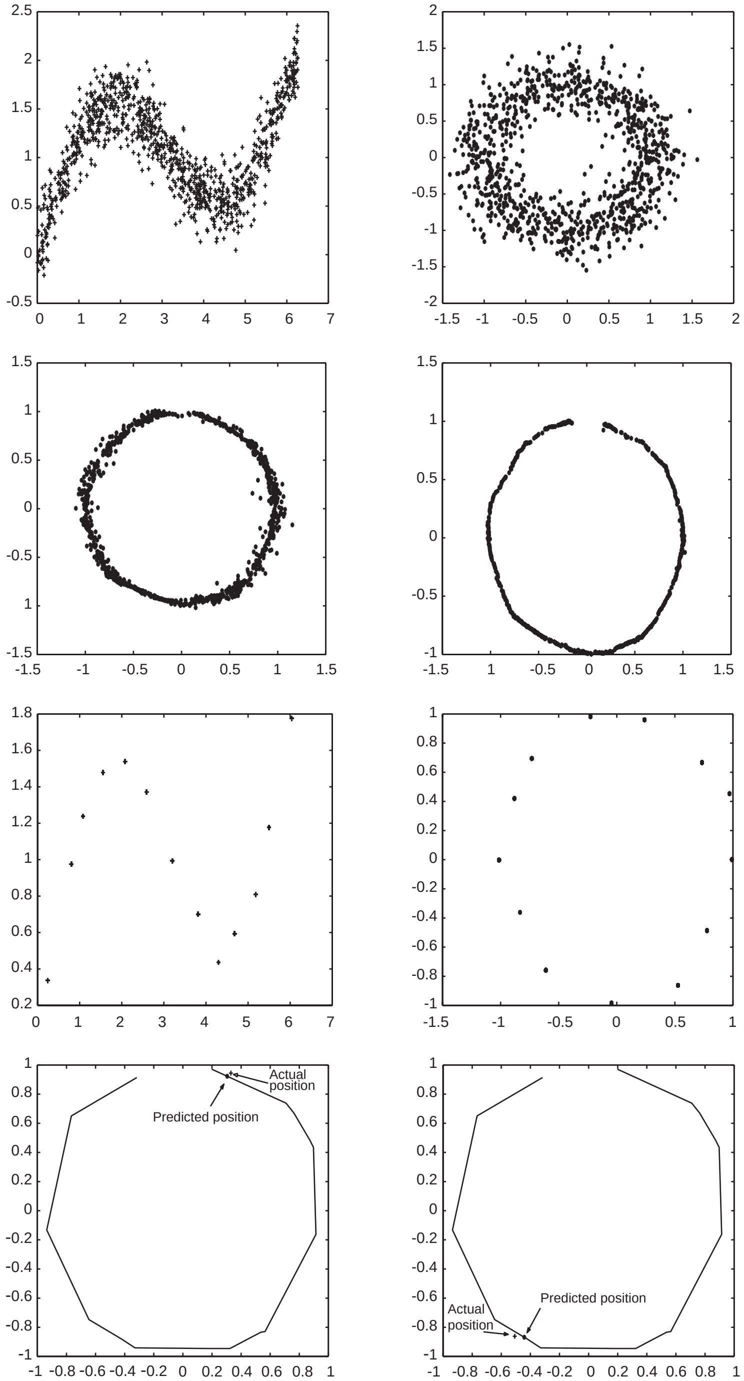
2.5
2
1.5
1
0.5
0
-0.5
0 1 2 3 4 5 6 7
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1.5 -1 -0.5 0 0.5 1 1.5 2
1.5
1
0.5
0
-0.5
1
-1.5
-1.5 -1 -0.5 0 0.5 1 1.5
1
0.5
0
-0.5
-1
-1.5 1 -0.5 0 0.5 1 1.5
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0 1 2 3 4 5 6 7
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1.5 -1 -0.5 0 0.5 1
Actual position
Predicted position
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1
Predicted position
Actual position
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1

Table 1
The first column gives the number of knot points and the others give the mean absolute percentage error on a test data set predicting 1 to 5 days ahead

Knot points	Day 1	Day 2	Day 3	Day 4	Day 5
57	1.0006	1.1086	1.2103	1.3035	1.4022
408	0.7413	0.9158	1.0685	1.1863	1.2887
607	0.6711	0.7939	0.9018	1.0197	1.0880

methods to find the underlying factors in this data set and then used a standard multilayered perceptron using backpropagation to predict each factor separately. To test our multilayered perceptron, we have split the data set into two sets: 1706 samples were used as the training data and 1706 for the test data. Each training input comprised a particular day's exchange rate plus the previous n days' exchange rates where values of n ranged from 5 to 25. With the current algorithm, we can simultaneously forecast as many days in advance as we wish, since our second principal curve can be as high dimensional as we wish. Typical results in terms of mean absolute percentage error on the test set are given in Table 1.

4. Twinned self-organising maps

Now the connection between principal curves and SOM has been discussed often in the literature, e.g. [13]. This suggests that the SOM might be used in a similar manner to the twinned principal curves algorithm. This may be conceptually thought of as two SOMs linked via the method of determining the winning neuron. Thus if the centres in our first space (the last 10 days data for example) are given by $\mathbf{w}_i$ and the centres in the second space (the 5 days ahead which we wish to predict) are given by $\mathbf{v}_i$, then we can select our winner using

$$c = \arg\min\{\|\mathbf{x}_1 - \mathbf{w}_i\| + \|\mathbf{x}_2 - \mathbf{v}_i\|\} \tag{1}$$

and then updating our individual centres each with the standard learning rules for a SOM.

$$\Delta\mathbf{w}_i = \eta\Lambda(c,i)(\mathbf{x}_1 - \mathbf{w}_i).$$

$$\Delta\mathbf{v}_i = \eta\Lambda(c,i)(\mathbf{x}_2 - \mathbf{v}_i),$$

where η is a learning rate and $\Lambda(c,i)$ is the neighbourhood function which in our case was a simple Gaussian. To test how accurate the trained model is on new data, we

←

Fig. 1. The top two diagrams show samples from the data sets. The middle diagrams show the first and second projections of the second data set. The third row shows the projections of both data sets after 10 iterations. The last row shows the results of forecasting the positions of points in data set 2 given only the position of the corresponding point in data set 1.

Table 2
Using the same data set of 3497 exchange rates, the mean absolute percentage error when using twinned SOMs

Centres	Day 1	Day 2	Day 3	Day 4	Day 5
25 (SOM)	0.6552	0.6671	0.6768	0.6908	0.6933

Table 3
The mean absolute percentage error when using the two methods to predict 88 students' open-book exams from their closed-book exams

SOM	0.1453	0.3055	0.3601
P Curve	0.2573	0.2494	0.1420

determine the winner only on the $\mathbf{x}_1$ data set

$$c = \arg\min\{\|\mathbf{x}_1 - \mathbf{w}_i\|\} \tag{2}$$

and use

$$|\mathbf{x}_2 - \mathbf{v}_c| \tag{3}$$

as a measure of the error of the prediction. Results on the same financial data set as previously are shown in Table 2 which should be compared with Table 1. We see that the SOM easily outperforms the twinned principal curves algorithm for this task for an equivalent number of centres or knot points.

4.1. Predicting student's exam marks

Our second experiment on a real data set uses data reported in [12]; it comprises 88 students' marks on 5 module exams. The exam results can be partitioned into two data sets: two exams were given as close book exams while the other three were opened book exams. The exams were on the subjects of Mechanics(C), Vectors(C), Algebra(O), Analysis(O), and Statistics(O). We thus split the five variables (exam marks) into two sets—the closed-book exams (x_{11},x_{12}) and the opened-book exams (x_{21},x_{22},x_{23}). One possible quantity of interest here is how highly a student's ability on closed-book exams is correlated with his ability on open-book exams. Alternatively, one might try to use the open-book exam results to predict the closed-book results (or vice versa). We have used the two methods above to attempt to predict the students' open-book exams from their closed-book exams. The results (in terms of mean absolute percentage errors) are shown in Table 3. The principal curve method has a slight advantage over the twinned SOM method but this advantage is reversed for Exam 3. It is very difficult to analyse why these results take the form that they do though clearly the results merit further study.

The twinned principal curve method used only 2 iterations through the data set while the twinned SOM method 100 000 samples (with replacement clearly) from the 88 data points.

5. Discussion

The twinned principal curves algorithm is a somewhat different algorithm from that suggested by [6] or [7] in that it iteratively uses a kernel smoother rather than attempting to approximate a principal curve by a mixture of straight lines. However it has a rather nice property of sparsification of the projections: the local averaging provides a smoothing of the data set and since we keep the values of ε_1 and ε_2 constant during the course of the simulation this smoothing progressively works out from each data point resulting in fewer and fewer projections onto the principal curve (compare the central two rows in Fig. 1). We may use this property to allow the number of distinct nodes we seek to determine the value of ε_1 and ε_2 (or vice versa).

It is worth noting also that this algorithm is able to deal with data sets which standard principal curve algorithms find difficult: the very fact of having two data sets with which to work simultaneously alleviates several problems. For example, since we initialise with a PCA and one of our data sets is circular, any diameter of the circle may be a principal component direction. This unfortunately means that points on opposite sides of the circle project onto the same part of the eigenvector and so we often have an initial twisting of the principal curve as it moves from the centre of mass on one side of the circle to the centre of mass on the other side, these centres of mass being caused by the finite numbers of samples. However, we only consider points to be local to the current point if they are local in both projections. This makes it much less likely that false neighbours will be chosen.

CCA maximises the correlation between two data sets under the constraint that the variance of $y_1 = \mathbf{w}_1^{\mathrm{T}}\mathbf{x}_1$ and $y_2 = \mathbf{w}_2^{\mathrm{T}}\mathbf{x}_2$ are both 1. Twinned principal curves can still meet this criterion; having found our sum of linear approximators, we may project new samples onto these twinned principal curves and calculate the variance of the resultant projections. In calculating new correlations, we may simply then divide each of y_1 and y_2 by their corresponding standard deviations.

We have been asked if the algorithm can be viewed as a single principal curve algorithm which has dimensionality equal to the sum of the dimensionality of $\mathbf{x}_1$ and $\mathbf{x}_2$. The answer is really no in that we use the criteria of closeness in each space independently and so simply having a single principal curve which joins together the two points $\mathbf{x}_1$ and $\mathbf{x}_2$ would give different results. It must be noted, however, that the SOM algorithm which does precisely this appears to work rather well.

5.1. Self-intersecting curves

One of the limiting factors for principal curves is that the curves (and hence the data set) should not intersect with itself. If this happens, the direction of maximum rate of change will not be unique at that point and so the principal curve cannot be found uniquely. However, when we have two data sets such intersections are permissible *provided* intersections in both data sets do not occur at the same time in both data sets.

Consider the data shown in the top line of Fig. 2, it comprises 2 sets of two-dimensional artificial data from $x_1(t) = \sin(t) + \mu_1, y_1(t) = \cos(t) + \mu_2, x_2(t) = t + \mu_3,$

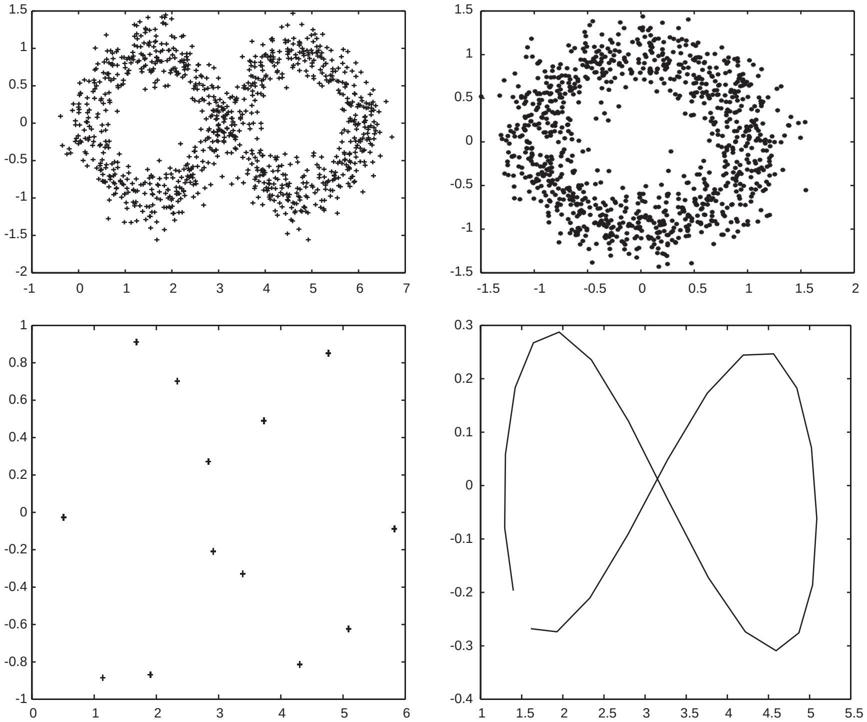

Fig. 2. The top two diagrams show samples from the data sets. The left diagram in the second row shows the projections of the intersecting data set after 10 iterations of the twinned principal curves algorithm; the right diagram shows a trained SOM's centres.

if $t \in [0, 2\pi]$ else $x_2(t) = (4\pi - t) + \mu_3, y_2(t) = t/3 + \sin(t) + \mu_4$ if $t \in [0, 2\pi]$ else $y_2(t) = (4\pi - t)/3 + \sin(4\pi - t) + \mu_4$ where t is drawn from a uniform distribution in $[0, 4\pi]$ and $\mu_i - N(0, 0.2)$ is Gaussian noise.

The knot points after 10 iterations of the twinned principal curves algorithm are shown in the second line of this figure.

We see that the method reliably finds the principal curves of both data sets. However it should be noted that if the first data point is chosen near the intersection point on the first curve, the second predicted data point can be very far from the correct second data point. The twinned SOM algorithm also is able to effectively model the intersecting data set as shown in Fig. 2.

5.2. The number of knot points

In the principal curve algorithm, we have a parameter (ε_1 or ε_2 respectively in our algorithm) which determines the width of the smoothing window and which thus eventually determines the number of knot points to which the algorithm converges. This also has an effect on the sum of squared errors (SSE) as shown in Table 4,

Table 4
The sum of the squared errors on a test set of artificial data

Iter	Knots	SSE	Knots	SSE	Knots	SSE
1	973	58.8	950	73.7	976	80.5
2	690	41.1	638	51.5	888	58.4
3	483	48.8	428	59.3	609	52.0
4	316	52.4	268	76.3	318	53.9
5	173	60.9	165	88.7	113	80.7
6	72	71.9	73	97.9	51	88.5
7	37	75.6	33	105.3	31	117.6

The first column gives the iteration number, the next two give the number of knot points and the sum of squared errors when $\varepsilon = 0.5$, the next two give the same information when $\varepsilon = 0.7$ and the last two the same information when $\varepsilon = 0.3$.

Table 5
Three simulations which show the varying number of knot points and the effect on mean absolulte percentage error when forecasting 1 day, 2 days, …, 5 days ahead

Knot points	Day 1	Day 2	Day 3	Day 4	Day 5
57	1.001	1.109	1.210	1.314	1.402
408	0.741	0.916	1.069	1.186	1.289
607	0.671	0.794	0.902	1.020	1.088

the first two columns are $\varepsilon = 0.5$, the second two with $\varepsilon = 0.7$ and the last two with $\varepsilon = 0.3$. In this section, we investigate criteria which may be used to decide what level of smoothing is optimal.

We see that as the number of knot points decreases, the error initially decreases before beginning to increase again. The later increase is due to an increase in bias in the learning machine—the number of knot points is not sufficient to adequately represent the data. The initial decrease is due to a decrease in variance as the noise is removed from the machine due to the smoothing effect of the algorithm.

Table 5 shows the effect of differing number of knot points on the mean absolute percentage error of forecasting 1 day, 2 days, …, 5 days ahead. We see that the lowest error is found when using the greatest number of knot points and that the error increases, for a given number of knot points, as we attempt to forecast further into the future. Table 6 shows the decreasing number of knot points in a simulation based on the dollar–pound exchange rate (3497 data points). We see that the number of knot points decreases to 18 and then remains stable for the last three iterations. The SOM algorithm has a width parameter which is generally pre-set and which decreases during the course of the simulation. The nearest equivalent to the algorithm here are those variants of the SOM algorithm which allow nodes to be dynamically added during the course of a simulation (e.g. [2]). We consider the problem of selecting the value of the smoothing parameter to remain an open question to which the results above can only point the way to an answer.

Table 6
The knot points (from a data set of 3497 exchange rates) after $1, 2, \ldots, 10$ iterations of the twinned principal curves algorithm

Knot points	Day 1	Day 2	Day 3	Day 4	Day 5
1459	0.907	1.008	1.128	1.216	1.328
915	0.920	1.025	1.145	1.237	1.346
571	0.935	1.037	1.162	1.248	1.360
297	1.044	1.143	1.252	1.335	1.430
134	1.194	1.278	1.373	1.441	1.539
42	1.255	1.328	1.421	1.491	1.579
25	1.330	1.397	1.489	1.548	1.632
18	1.349	1.416	1.504	1.566	1.644
18	1.349	1.416	1.504	1.566	1.644
18	1.349	1.416	1.504	1.566	1.644

The simulation converges to a stable 18 knot points.

6. Conclusion

We have shown that the principal curve method can be extended to work on two data sets simultaneously and that using two data sets is, in fact, advantageous in that there is less chance of two projections simultaneously misleading than there is of a single projection being misleading. Also when we use this algorithm to forecast, we have the advantage that it is very simple to forecast a number of days ahead simultaneously: this simply increases the dimensionality of the space through which the second principal curve moves. The results from the foreasting were comparable to that from our previous methods [4] and were considerably easier to achieve: we performed no optimisation to get the reported results and found comparable results over a wide range of parameter values.

However the known similarity between principal curves and SOMs suggested the twinned SOMs algorithm and this was shown experimentally to outperform the twinned principal curve algorithm in terms of minimisation of the mean absolute percentage error on the financial data set. Of course this does not mean that the twinned SOM algorithm will do better in every task and this is the topic for future research and, in fact, on the task of forecasting students' marks on one set of exams from those in another set of exams, the twinned principal curve method performed best.

Finally, we consider that the task of forecasting may not be the best task for either of these methods: the SOM is often most keenly appreciated when used as an aid to visualising structure in high dimensional data sets and the principal curves algorithm is also prominent in this field. Thus, one topic for future research will be a comparison of these methods when used for visualisation.

References

[1] P. Delicado, Another look at principal curves and surfaces, J. Multivariate Anal. 77 (2001) 84–116.

[2] B. Fritzke, Kohonen feature maps and growing cell structures—a performance comparison, in: Advances in Neural Information Processing Systems 5, Morgan Kaufmann, Los Altos, CA, 1993.

[3] Z. Gou, C. Fyfe, A family of networks which perform canonical correlation analysis, Internat. J. Knowledge-based Intelligent Eng. Systems 5 (2) (2001) 76–82.

[4] Y. Han, C. Fyfe, Finding underlying factors in time series, Cybernet. Systems: An Internat. J. 33 (2002) 297–323.

[5] Y. Han, C. Fyfe, Forecasting using twinned principal curves, in: Tenth European Symposium on Artificial Neural Networks, ESANN2002, April 2002.

[6] T. Hastie, W. Stuetzle, Principal curves, J. Amer. Statist. Assoc. 84 (406) (1989) 502–519.

[7] B. Kegl, A. Krzyzak, T. Linder, K. Zeger, Learning and design of principal curves, IEEE Trans. Pattern Anal. Mach. Intell. 22 (3) (2000) 281–297.

[8] T. Kohonen, Self-Organising Maps, Springer, Berlin, 1995.

[9] P.L. Lai, C. Fyfe, A neural network implementation of canonical correlation analysis, Neural Networks 12 (10) (1999) 1391–1397.

[10] P.L. Lai, S.J. Chuang, C. Fyfe, Power load forecasting using neural canonical correlates, in: International Conference on Pattern Recognition, ICPR'2000, IEEE Cs Press, 2000.

[11] P.L. Lai, C. Fyfe, Kernel and nonlinear canonical correlation analysis, Internat. J. Neural Systems 10 (5) (2001) 365–377.

[12] K.V. Mardia, J.T. Kent, J.M. Bibby, Multivariate Analysis, Academic Press, New York, 1979.

[13] F. Mulier, V. Cherkassky, Self-organisation as an iterative kernel smoothing process, Neural Comput. 6 (6) (1995) 1165–1177.