ESANN 2010 proceedings, European Symposium on Artificial Neural Networks - Computational Intelligence and Machine Learning. Bruges (Belgium), 28-30 April 2010, d-side publi., ISBN 2-930307-10-2.

Distance Functions for Local PCA Methods

Alexander Kaiser, Wolfram Schenck, and Ralf Möller[*]

Computer Engineering Group — Faculty of Technology
Bielefeld University — POB 100131, D-33501 Bielefeld — Germany
akaiser'at'ti.uni-bielefeld.de

Abstract. The NGPCA method, a combination of the robust neural gas vector quantization method and a fast neural principal component analyzer, has proved to be a valuable tool for the generalized learning of high–dimensional data. At its core, the method uses a competitive ranking to adapt its units. The competition is guided by a specialized distance function — known as the normalized Mahalanobis distance — that assumes elliptic cluster shapes. Recently, an alternative distance function, the normalized Rayleigh quotient, has been suggested. This paper compares the performance of NGPCA on different distance functions. For the comparison a data set from a realistic robot arm experiment is used.

1 Introduction

Principal component analysis (PCA) is an ubiquitous statistical method for dimension reduction. Although PCA is linear in its nature, it has been successfully applied to a large variety of problems ranging from data compression to pattern recognition. PCA is based on a linear projection from a high–dimensional data space into a low–dimensional feature space, yielding the so–called *principal components*. The projection is information–preserving in the sense that it takes into account those directions in which the data have their highest variance and thus convey most of their information. The other directions which are considered irrelevant are dismissed which eventually leads to a loss of information. The distance between a data point and its reconstruction from its corresponding principal components, which is known as the *reconstruction error*, is the quantity which is sought to be minimized (usually over all data points). The projection which fulfills this optimality condition is the projection into the subspace spanned by the eigenvectors corresponding to the m largest eigenvalues of the sample covariance matrix.

The objective of local PCA methods [1, 2] is to extend the basic PCA approach by augmenting it with a vector quantization (VQ) method. This allows for the accurate approximate of a globally nonlinear structure (e.g. a curved manifold) by locally applying linear PCA. Commonly, local PCA approaches are based on specialized distance functions. Among these are the normalized Mahalanobis distance [2] and the normalized Rayleigh quotient [3]. The choice of the distance function has a strong influence on the performance of the local PCA method. The objective of this paper is to review two of these distance

[*]This work was supported by the German Science Foundation (DFG) under grants MO 1037/1-2 and EXC277 (Excellence Cluster "Cognitive Interaction Technologies").

functions and to evaluate their performance when used in an on–line local PCA framework which will be presented in the following.

2 Local PCA method

In this section we will briefly review a local PCA method, known as *neural gas principal component analysis* (NGPCA) [2]. This method is an on–line learning algorithm. The algorithm is an extension of the *neural gas* (NG) algorithm [4] to local PCA. In contrast to other VQ algorithms, neural gas uses a soft competition between prototypes during the training. The goal of this heuristic is to prevent the algorithm from getting stuck in local minima [4].

2.1 NGPCA

The extension of NG to local PCA requires two major modifications: The modified code book is a set of units which consist of the center and the local PCA subspace which is represented by an $n \times m$ matrix $\boldsymbol{W}_i$ containing the m principal eigenvectors, an $m \times m$ diagonal matrix $\boldsymbol{\Lambda}_i$ containing the m principal eigenvalues, and the residual variance σ_i^2 which is the sum of the $n-m$ minor eigenvalues. These entities form a 4–tuple $U_i = (\boldsymbol{c}_i, \boldsymbol{W}_i, \boldsymbol{\Lambda}_i, \sigma_i^2)$, to which we will refer as a unit.

Let the code book be denoted by $\mathcal{U} = \{U_1, \ldots, U_N\}$. The assignment between units and data points is calculated using the distance function

$$d(\boldsymbol{x}, U_i) = \boldsymbol{y}_i^T \boldsymbol{\Lambda}_i^{-1} \boldsymbol{y}_i + \frac{1}{\lambda_i^*}(\|\boldsymbol{\xi_i}\|^2 - \|\boldsymbol{y}_i\|^2) + \ln|\boldsymbol{\Lambda}_i| + (n-m)\ln(\lambda_i^*), \tag{1}$$

where $\boldsymbol{\xi}_i = \boldsymbol{x} - \boldsymbol{c}_i$ denotes the deviation between $\boldsymbol{x}$ and the unit center, $\boldsymbol{y}_i = \boldsymbol{W}_i^T \boldsymbol{\xi}_i$ denotes the vector of principal components, and $\lambda_i^* = \frac{\sigma_i^2}{n-m}$ is the estimate of the minor eigenvalues. Furthermore, $|\boldsymbol{\Lambda}_i| = \prod_{j=1}^m \lambda_i^{(j)}$ denotes the determinant of $\boldsymbol{\Lambda}_i$. The distance function (1) is called *normalized Mahalanobis distance.* In fact, the iso–distance surface of (1) are hyper–ellipsoids, centered at $\boldsymbol{c}_i$ and augmented with a hyper–sphere in the $n-m$ minor dimensions. The term $E_{\text{recon}} = \|\boldsymbol{\xi_i}\|^2 - \|\boldsymbol{y}_i\|^2$, which also appears in (1), is equivalent to the instantaneous reconstruction error. The term in (1) which depends on the determinant of $\boldsymbol{\Lambda}_i$ is related to the volume of the hyper–ellipsoid; this term penalizes “large” hyper–ellipsoids during the training [2].

For every time step t, an input vector $\boldsymbol{x}(t) \in \mathcal{X} = \{\boldsymbol{x}(1), \ldots, \boldsymbol{x}(T)\} \subset \mathbb{R}^n$ arrives, and the distance between $\boldsymbol{x}(t)$ and each unit is calculated using (1). Each unit is then assigned a rank r_i according to its distance; the unit which has the lowest distance gets rank 0, the second lowest gets rank 1, etc. The rank is then used to calculate an individual learning rate for each unit:

$$\alpha_i = \epsilon(t) \exp(-r_i/\rho(t)), \tag{2}$$

where r_i denotes the rank, $\rho(t)$ denotes the neighborhood range, and $\epsilon(t)$ is a global learning rate. The parameters $\rho(t)$ and $\epsilon(t)$ both depend on t and decay

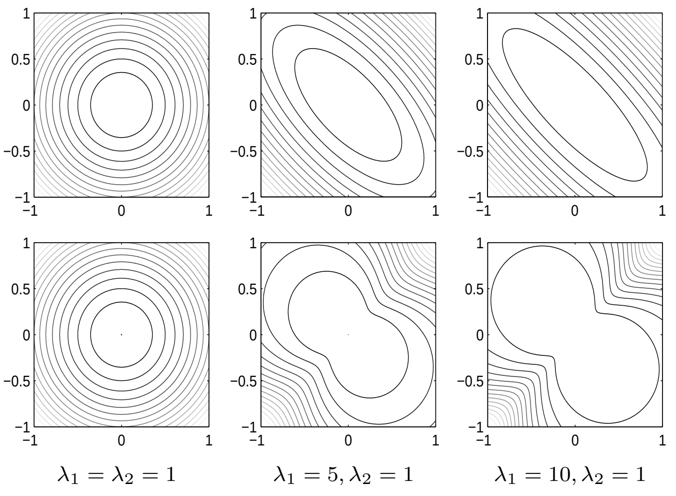

Fig. 1: 2D contours of the normalized Mahalanobis distance (top row), and the reciprocal normalized Rayleigh quotient (bottom row) for $\lambda_1 \in \{1, 5, 10\}$ and $\lambda_2 = 1$.

exponentially during the course of the training from their initial values $\rho(0), \epsilon(0)$ to their (small) final values $\rho(T), \epsilon(T)$ [4, 2].

In the following, we briefly summarize the learning rule of NGPCA. The adaptation of the unit centers is analog to NG [4]:

$$\boldsymbol{c}_i(t+1) \leftarrow \boldsymbol{c}_i(t) + \alpha_i \cdot (\boldsymbol{x}(t) - \boldsymbol{c}_i(t)), \tag{3}$$

The subspaces are adapted by an on–line PCA method [5]. We write this step abstractly as

$$\boldsymbol{W}_i(t+1), \boldsymbol{\Lambda}_i(t+1) \leftarrow \mathrm{PCA}(\alpha_i, \boldsymbol{W}_i(t), \boldsymbol{\Lambda}_i(t), \boldsymbol{\xi}_i). \tag{4}$$

The residual variance — which is equal to the mean reconstruction error — can be calculated recursively, using the update equation

$$\sigma_i^2(t+1) \leftarrow \sigma_i^2(t) + \alpha_i(\|\boldsymbol{\xi}_i(t)\|^2 - \|\boldsymbol{y}_i(t)\|^2 - \sigma_i^2(t)). \tag{5}$$

Initially, the unit centers are placed by picking samples from the data distribution at random, the eigenvectors are initialized with an arbitrary orthonormal set of vectors, and the eigenvalues and the residual variance are initially set to common values $\lambda(0)$ and $\sigma^2(0)$, respectively.

2.2 Alternative distance function

Besides the normalized Mahalanobis distance (1), there are other distance functions that are especially suited for local PCA [2, 3]. The distance function we

	func.	$N = 60$	$N = 120$
$m = 3$	Mahalanobis	**1.36 (0.11)**	1.18 (0.08)
	Rayleigh	2.04 (0.23)	1.61 (0.17)
	Euclidean	1.84 (0.27)	**1.10 (0.06)**
$m = 9$	Mahalanobis	5.53 (1.15)	3.01 (0.72)
	Rayleigh	**1.01 (0.13)**	**0.78 (0.07)**
	Euclidean	1.19 (0.13)	0.87 (0.12)

Table 1: Grasping error E_{grasp} for the different distance functions (standard deviation in brackets).

will focus on in the following is motivated by one special drawback of NGPCA (and presumably other local PCA methods): During the training, units might be placed (under certain conditions) in a way that they are not supported by any of the data. Therefore, no data points are assigned to these units — termed "dead" units —, and thus they do not contribute to the approximation of the data distribution.

Huang et al. [3] propose a distance function — the normalized Rayleigh quotient — which they claim is independent of the size or volume of the hyper–ellipsoid, and thus results in less "dead" units [3]. We will present a modified variant of the normalized Rayleigh quotient which is better suited for the NG-PCA framework. First of all, we replaced the full covariance matrix with its eigendecomposition in analogy to (1). This avoids the costly update of the covariance matrix which can be very large for high–dimensional data. Another modification was necessary, because the normalized Rayleigh quotient is subject to maximization whereas a distance function should be minimized. Therefore, we take the reciprocal of the original equation [3], yielding

$$\tilde{d}_R(\boldsymbol{x}, U_i) = \frac{(\operatorname{tr} \boldsymbol{\Lambda}_i + (n-m)\lambda_i^*) \cdot \|\boldsymbol{\xi}_i\|^4 + \epsilon_r}{\boldsymbol{y}_i^T \boldsymbol{\Lambda}_i \boldsymbol{y}_i + \lambda_i^* (\|\boldsymbol{\xi}_i\|^2 - \|\boldsymbol{y}_i\|^2) + \epsilon_r}, \tag{6}$$

where ϵ_r is a small regularizing constant (in our experiments $\epsilon_r = 10^{-6}$) that prevents (6) from becoming undefined if $\|\boldsymbol{\xi}_i\| = 0$.

Figure 1 (top row) shows the contours of (1) for different λ_1 and fixed λ_2. The contours are ellipses whose major axes vary along with the principal eigenvalue λ_1. In contrast, figure 1 (bottom row) shows the contours of (6) which only vaguely resemble hyper–ellipsoids. Although the extension in the principal direction is mainly affected by the principal eigenvalue, it has also a strong influence on the surface's extension in its minor direction. The two bulges at each side of the surface become more prominent as the ratio λ_1/λ_2 increases.

3 Results

We applied NGPCA to data from a real robot arm experiment [6]. The robot setup consist of two cameras, each of them mounted on a pan–tilt unit, and

	func.	$N = 60$	$N = 120$
$m = 3$	Mahalanobis	2.60 (1.20)	20.00 (3.38)
	Rayleigh	7.20 (1.94)	33.10 (2.84)
	Euclidean	**0.00 (0.00)**	**3.10 (1.64)**
$m = 9$	Mahalanobis	25.90 (1.87)	70.80 (3.84)
	Rayleigh	9.60 (2.50)	37.20 (4.42)
	Euclidean	**0.10 (0.30)**	**4.30 (1.62)**

Table 2: Number of "dead" units. See text for explanations.

a 6–degrees–of–freedom robot manipulator with gripper. The setup is facing a table with a colored wooden block on it. A saccade controller directs the gaze of the cameras towards the block. The task is now to learn an association between the gaze direction, the visual input, and an arm posture suitable for grasping the block.

The pattern space is divided into 20 input dimensions which encode the position/orientation of the block, and 48 output dimensions which correspond to the grasping posture of the arm, resulting in a total of 68 dimensions. The data set consists of 3200 distinct patterns 85% of which were used for training and 15% for testing. All results presented in the following were obtained using the test set.

We use NGPCA as an abstract recurrent neural network [6] to recall for a given input (i.e. the gaze direction and the block's orientation) a given output (i.e. a grasping posture). We did not succeed in finding a closed–form solution of the recall method for the normalized Rayleigh quotient yet. Therefore, we used the recall method in its original form [6] which is based on the normalized Mahalanobis distance.

In our experiments, the arm is not moved physically. We rather use its forward kinematics to calculate the position/orientation of the gripper at the recalled grasping posture. These values are compared to the position/orientation of the block which yields four errors: The horizontal and vertical position error and the horizontal and vertical orientation error. These errors were combined as a weighted sum into a single index which we will refer to as the grasping error E_{grasp}.

For the comparison of the different distance functions (see section 2), we calculated the mean grasping error and the mean number of dead units, respectively. We compared the normalized Mahalanobis distance (1), the normalized Rayleigh quotient (6), and the Euclidean distance. Networks were trained using $N = 60$ and $N = 120$ units, and $m = 3$ and $m = 9$ eigenvectors, respectively. For each parameter combination and distance function, 10 NGPCA networks were trained. The number of training steps was $T = 30\,000$. The learning rate and neighborhood radius were exponentially decreased from $\epsilon(0) = 0.5$ and $\rho(0) = 1$ to $\epsilon(T) = 0.01$ and $\rho(T) = 0.01$, respectively, during the course of the training. The eigenvalues and the residual variance were set to a common initial value of

1. Furthermore, we used the robust recursive least squares learning algorithm (RRLSA) [5] for the on–line PCA learning.

Table 1 shows the results (E_{grasp}) for the different distance functions. Obviously, the normalized Mahalanobis distance is inferior in most of the cases; it is only superior for $N = 60, m = 3$. The Euclidean distance is otherwise the best for three eigenvalues. The normalized Rayleigh quotient, which shows the worst performance for $m = 3$, clearly outperforms the normalized Mahalanobis distance for $m = 9$, but is only slightly better than the Euclidean distance.

Table 2 shows the mean number of "dead" units. It is obvious that the occurrence of "dead" units scales with the total numbers of units, although the strength of this effect varies for the different functions. Moreover, the normalized Mahalanobis distance is very sensitive to the choice of m: For $m = 3$ the number of dead units is much smaller than for $m = 9$. The Euclidean distance and the normalized Rayleigh quotient are widely unaffected by the choice of m.

4 Conclusion

The NGPCA method is a local PCA method whose core concept is a competition between the local PCA units (ranking) that relies on a specialized distance function. Two of these distance functions — the normalized Mahalanobis distance and the normalized Rayleigh quotient — were reviewed and their performance was evaluated in a comparative study on data from a robotic arm experiment.

The normalized Mahalanobis distance performs well if the intrinsic dimensionality of the data distribution is known beforehand. However, the more the dimensionality is overestimated, the worse its performance becomes. The drop in performance is indicated by a higher grasping error and an increase of the occurrence of "dead" units. The normalized Rayleigh quotient fares better if the number of eigenvectors is increased. Futhermore, if the number of units is relatively large, there is no advantage in using a specialized distance function and one may use the Euclidean distance instead.

References

[1] N. Kambhatla and T. K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.

[2] Ralf Möller and Heiko Hoffmann. An extension of neural gas to local PCA. *Neurocomputing*, 62(1):305–326, December 2004.

[3] Dong Huang, Zhang Yi, and Xiaorong Pu. A new local PCA-SOM algorithm. *Neurocomputing*, 71(16-18):3544–3552, October 2008.

[4] Thomas M. Martinetz, Stanislav G. Berkovich, and Klaus J. Schulten. "Neural-gas" network for vector quantization and its application to time-series prediction. *IEEE Trans. Neural Networks*, 4(4):558–569, July 1993.

[5] Shan Ouyang, Zheng Bao, and Gui-Sheng Liao. Robust recursive least squares algorithm for principal component analysis. *IEEE Trans. Neural Networks*, 11(1):215–221, January 2000.

[6] Heiko Hoffmann, Wolfram Schenck, and Ralf Möller. Learning visuomotor transformations for gaze-control and grasping. *Biological Cybernetics*, 93(2):119–130, 2005.