An Iterative Approach to local-PCA

Samuel John, Heiko Wersing and Helge Ritter

***Abstract*— We introduce a greedy algorithm that works from coarse to fine by iteratively applying localized principal component analysis (PCA). The decision where and when to split or add new components is based on two antagonistic criteria. Firstly, the well known quadratic reconstruction error and secondly a measure for the homogeneity of the distribution. For the latter criterion, which we call "generation error", we compared two different possible methods to assess if the data samples are distributed homogeneously. The proposed algorithm does not involve a costly multi-objective optimization to find a partition of the inputs. Further, the final number of local PCA units, as well as their individual dimensionality need not to be predefined. We demonstrate that the method can flexibly react to different intrinsic dimensionalities of the data.**

I. Introduction

Many problems in machine learning are related to finding the low-dimensional manifold which is spanned by the data samples. Psychophysical and biological findings suggest that perceptual discrimination among other tasks are performed on a low-dimensional representation of the sensory inputs [2]. One family of methods – the vector quantization (VQ) approaches – aim to approximate the density by positioning point-like representatives in regions of high density, like the Growing Neural Gas (GNG) by [5]. An important extension is to build a lower dimensional map, that represents the neighborhood relation of the representatives on the manifold. The self organizing map (SOM)[4], and local linear embedding (LLE)[9] are examples of that latter kind. Another family replaces the point-like elements by local principal component analysis (PCA)[3] that represents directions of high variance in the data. The benefit is that more sample points can be described by a few parameters (the position on the local PCA-coordinate systems) instead of filling a hyper-volume with increasingly many representative points. However, when going from point-like representatives to local coordinate systems some additional problems are introduced. One difficulty is that the dimensionality of the local-PCA often must be equal for all elements and it is difficult to estimate a fixed number of dimensions that matches the intrinsic dimensionality of the data well. There are estimation techniques[1] that can address the overall intrinsic dimensionality, but not all manifolds show consistently the same local dimensionality and it is likely to observe areas of different local dimensionality in real world data.

Samuel John (sjohn@cor-lab.uni-bielefeld.de) is doing his PhD at the Cognition and Robotics-Lab (CoR-Lab.de), Bielefeld University, PO Box 10 01 31, D-33501 Bielefeld, Germany and the Honda Research Institute Europe GmbH

Dr. Heiko Wersing (heiko.wersing@honda-ri.de), Honda Research Institute Europe GmbH, Carl-Legien-Str. 30, 63073 Offenbach/Main, Germany

Prof. Dr. Helge Ritter (helge@techfak.uni-bielefeld.de), Coordinator Excellence Cluster 277: Cognitive Interaction Technology Director Cognition and Robotics Laboratory (CoR-Lab) Faculty of Technology, Bielefeld University, 33501 Bielefeld, Germany

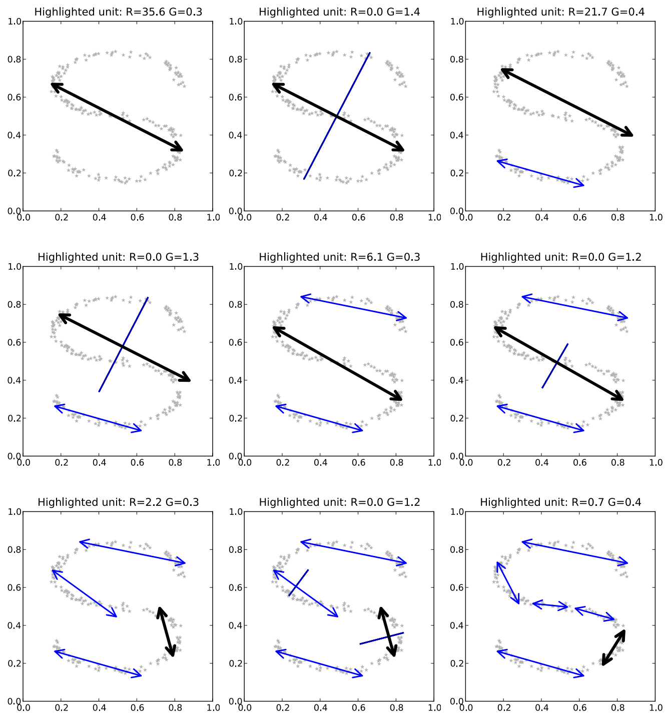

Fig. 1. ***Iterative refinement.*** These plots show the progression of our algorithm (with the generation error as defined in equation 4), until the error thresholds are met. 200 data samples are drawn from a noisy-"S" distribution. The first principal component is drawn as an arrow and the second (if available) as a line, which crosses the principal one. For this example the iteration stops at step nine. The threshold for the reconstruction error t_R for each element is set to 2.0 and the threshold for the generation error t_G is 0.8. Starting from a global view (top-left), the points are seen as a two-dimensional area in step two (top-middle), towards a successively more structured "chaining" of several one-dimensional localized elements. In each image, one of the PCA-elements is printed in bold and the reconstruction error E_R and the generation error E_G for that highlighted component are noted above each plot (R$\hat{=}E_R$ and G$\hat{=}E_G$). For example, in the fifth step, the generation error is quite low (G=0.3), but the reconstruction error (R=6.1) is still higher than the threshold of 2.0. This leads to increasing the dimension of that local-PCA element in step six, where the reconstruction error falls to zero, but the generation error actually increases (G=1.2).

Another common difficulty of local PCA methods is that the partition of the input space into separated areas is decoupled from the PCA computation and several approaches have been suggested to optimize both at the same time; the partition and the error introduced by projecting the data on the principal components [6], [7].

In this manuscript we present a new greedy approach to the family of local PCA methods. Our algorithm circumvents both before-mentioned difficulties by using two opposing "forces": A local squared reconstruction error and a measure

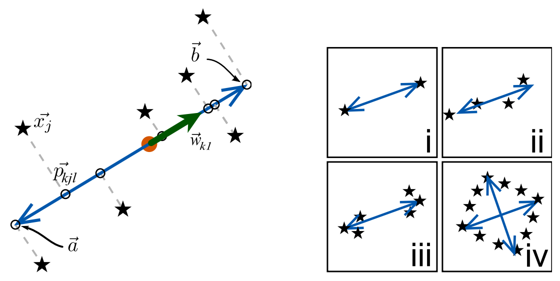

Fig. 2. ***Left: Generation of an axis for a localized coordinate system.*** On the given a number of samples $\vec{x}_j \in X$, a PCA is is computed. The principal eigenvector is $\vec{w}_{k1}$ and the line extends from the minimum to the maximum of the projections onto that direction.
Right: When is a line a sufficiently good approximation? For the cases (i) and (ii) we may find that the points are represented quite well. For the pictures (iii) and (iv), there may be more "structure" in the data than captured by a single line segment or an additional representation dimension (iv). From the viewpoint of the generation error, however, only (ii) may be appropriately generated from the model.

for the homogeneity. It is well known that the principal component analysis minimizes the squared reconstruction error. By increasing the number of principal components, the reconstruction error (often dramatically) drops. In the extreme, the number of components is equal to the number of dimensions of the input space.

An appropriate description of a data manifold should incorporate the idea of a generative model in the sense that new samples, which can be generated by choosing arbitrary positions in the space, spanned by the coordinate system of a local PCA, are located in the vicinity of the manifold. We call the error of generating samples that do not belong (or are not close to) the underlying data manifold "generation error". Clearly, a good approximation to a given data-set should possess a small reconstruction error and – at the same time – a small generation error. It is not at all obvious how to assess the generation error of the space spanned by a local PCA and therefore we tested and compared two possible alternative candidates. One measure we studied, uses the chi-square-test to check whether the empirical (local) distribution of training data can be considered to stem from a uniform distribution[8]. The alternative – which, interestingly, is much better suited – simply measures the gradient of the empirical distribution from the axes of the local coordinate system "outwards" to the regions "far" from the axes. The latter measure assumes a centered distribution density with a falling density towards the borders of the local region. Certain distributions with non-homogeneous densities, for example a ring-like structure, are typical cases where the PCA – in its nature of a linear method – fails to capture the essence of the data. On these distributions, the gradient from the center of the PCA coordinate axes is not monotonously falling.

II. Model

The fundamental assumption in the proposed method is that most samples lie close to a lower-dimensional manifold, which is embedded in the input space and only few (or no) samples lie far off. Therefore, the coordinate systems – defined by local-PCA – should lie in or along local maxima of the empirical density. Based on the reconstruction error and the generation error, we iteratively and strictly locally decide whether to increase the dimensionality or break up a local region into two parts. For each of the sub-regions, the algorithm is applied anew.

A. Local PCA coordinate systems

We describe a local-PCA unit on the subset X_k of all input samples I with $X_k \subset I \subset \mathbb{R}^n$ and $N(X_k)$ elements by the tuple

$$A_k = \{\vec{\mu}_k, d_k, \mathbf{W}_k, \mathbf{\Lambda_k}\}$$

where k is the index of the local-PCA unit. The vector $\vec{\mu}_k$ describes the center of the PCA and the number $d_k \in \mathbb{N}$ represents the dimensionality of the PCA. The $n \times d_k$ Matrix $\mathbf{W}_k$ encodes the first d_k principal eigenvectors $\vec{w}_{kl}$ (of the data-points in the set X_k) in its columns $l = 1, ...d_k$. The $d_k \times d_k$ diagonal matrix $\mathbf{\Lambda}_k$ contains the corresponding eigenvalues $\lambda_{k1}, ..., \lambda_{kd_k}$. The computation of the d_k principal components of X_k is denoted by PCA(d_k, X_k) and yields such a tuple A_k. A point from the set X_k is written as $\vec{x}_j$.

We describe the l-th axis $l \in \{1, ..., d_k\}$ of the local coordinate system by a starting-point $\vec{a}_{kl}$ and an end-point $\vec{b}_{kl}$. The projection onto an axis is written as

$$P_{kl}(\vec{x}_j) = \langle \vec{x}_j - \vec{\mu}_k, \vec{w}_{kl} \rangle$$

where $\langle \cdot, \cdot \rangle$ denotes the scalar product. For the given samples $\vec{x}_j \in X_k$, the axis extends from the minimum to the maximum in the direction of the l-th principal component (See Figure 2).

$$\vec{a}_{kl} = \vec{\mu}_k + \vec{w}_{kl} \min_j (P_{kl}(\vec{x}_j))$$
$$\vec{b}_{kl} = \vec{\mu}_k + \vec{w}_{kl} \max_j (P_{kl}(\vec{x}_j))$$

An input $\vec{x}_j$ is reconstructed by $\vec{p}_{kj}$ with[1]

$$\vec{p}_{kj} = \vec{\mu}_k + \sum_{l=1}^{d_k} \vec{w}_{kl} P_{kl}(\vec{x}_j) \tag{1}$$

in the coordinates of the whole input space.

B. Overall algorithm

The main iteration, to find a partition of the input space, as well as the dimensionality and the number of local PCA units, is described by the pseudo-code in algorithm 1, where $E_R(\cdot)$ and $E_G(\cdot)$ are the two different error measures, described in section II-C and II-D. For each unit A_k, the errors $E_R(A_k)$ and $E_G(A_k)$ must fall below the thresholds t_R and t_G respectively for the algorithm to stop. The function Split(A_k) returns two new tuples $\{A_p, A_q\}$ and thereby implicitly partitions the set X_k into two subsets X_p and X_q as described in section II-E.

[1] In order to keep $\vec{p}_{kj}$ between $\vec{a}_{kl}$ and $\vec{b}_{kl}$ for all l, an additional clipping operation is performed if $\vec{p}_{kj}$ would exceed the ends of one axis.

Algorithm 1 Greedy iteration.

```
    $X_0 \leftarrow I$
    $A_0 \leftarrow$ PCA$(1, X_0)$
    $K \leftarrow \{A_0\}$  # K is a set
    finished ← False
    while not finished:
        finished ← True
        For each $A_k$ in $K$:
            $L \leftarrow \{\}$  # empty set
            If $E_R(A_k) > t_E$:
                $L \leftarrow L \cup \{$PCA$(d_k + 1, X_k)\}$
                finished ← False
            Else if $E_G(A_k) > t_G$:
                $L \leftarrow L \cup$ Split$(A_k)$
                finished ← False
            Else:
                $L \leftarrow L \cup \{A_k\}$
        $K \leftarrow L$
```

The iteration runs until no A_k is changed any more and can be summarized by: "Increasing the dimensions of the local PCA units until the reconstruction error is small enough, and then, if the generation error is too high, splitting it into two one-dimensional local-PCA to start over again with increasing the dimensions."

C. Reconstruction Error

The reconstruction error $E_R(A_k)$ on the k-th unit is measured on the samples $\vec{x}_j \in X_k$ that belong to A_k, with $N(X_k)$ denoting the number of samples:

$$E_R(A_k) = \sum_{j=1}^{N(X_k)} \frac{\|\vec{x}_j - \vec{p}_{kj}\|^2}{N(X_k)}$$

Obviously, with an increasing number of dimensions d_k, this error gets smaller and eventually reaches zero. So, whenever the reconstruction error for an A_k is above the target t_R, increasing the number of dimensions is a reasonable choice.

D. Generation Error

As explained in the introduction, the choice of the generation error is not a priori clear. We studied two alternatives:

For the first candidate to measure the generation error $\widetilde{E}_G$, we assume that the the density of the line, rectangle, cuboid or hyper-cuboid that is covered by the axes of a local-PCA unit A_k can be considered as a uniform distribution. We use the chi-square test [8] to get an approximate measure, how large the empirical distribution deviates from a uniform distribution. A d_k-dimensional grid G_k is computed to represent the empirical density by adding $+1$ into the closest bin for each sample $\vec{x}_k$. The grid has h bins in each dimension, that is h^{d_k} bins in total (Figure 3). The one-dimensional helper function for the binning is named $\text{bin}_{kl}(\cdot)$ and it returns the index of a bin along the l-th axis A_k. The number of categories for the test is the total number of bins h^{d_k}. For reasons of simplicity, we write G_k^i to denote the value of the

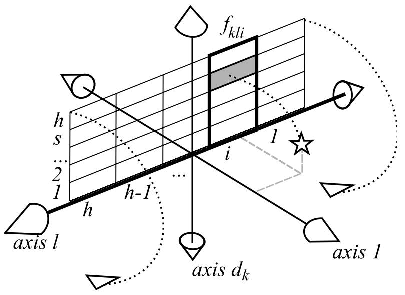

Fig. 3. ***Illustration of the histogram over the distances for each axis.*** For each axis $1, ..., l, ..., d_k$ a two-dimensional histogram is computed with h times h bins. This histogram is always two-dimensional and does notchange with the dimensionality of the input space. One can think of this as rotating a grid around an axis l and each input (star-symbol) is put in the corresponding bin (i, s). The functions f_{kli} describe the i-th column of the histogram and $H_{kl}(i)$ (See equation 3) sums over the differences along the i-th column.

i-th bin just as if we would iterate over a all h^{d_k} bins as a flat list, no matter of the number of axes.

$$\widetilde{E}_G = \chi^2 = \sum_{i=0}^{h^{d_k}} \frac{\left(G_k^i - \frac{N(X_k)}{h^{d_k}}\right)^2}{\frac{N(X_k)}{h^{d_k}}} \tag{2}$$

The second candidate for E_G implies a much weaker assumption by looking only at the distances of the data points towards an axis. In contrast to the d_k dimensional grid, we compute a two-dimensional histogram for each axis as illustrated in figure 3. If the histogram over the distances increases, it means that more samples are far away than near the axis. In other words, if the axis is positioned approximately in or along an area of local maximal density, the histogram decreases. For each axis $l \in \{1, ..., d_k\}$, we measure the distances of the reconstructed sample $\vec{p}_{kj}$ to the corresponding point $\vec{p}_{kjl}$ projected only onto the l-th axis. Therefore, we first define a one-dimensional histogram for the distances with h bins and a bin-width of $u_{kl} = \frac{\max_j(\|\vec{p}_{kj} - \vec{p}_{kjl}\|)}{h}$ by

$$f_{kli}(s) = \sum_{j=1}^{N(X_k)} \begin{cases} 1 \text{ if } \text{bin}_{kl}(\vec{x}_j) = i \wedge \left\lfloor \frac{\|\vec{p}_{kj} - \vec{p}_{kjl}\|}{u_{kl}} + \frac{1}{2} \right\rfloor = \left\lfloor s + \frac{1}{2} \right\rfloor \\ 0 \text{ else} \end{cases}$$

and i means the i-th bin along the axis l. The argument $s = 1, ..., h$ marks the bin for the distance towards the axis. This histogram f_{kli} is evaluated for $i = 1, .., h$ equally distributed bins along the l-th axis. And, as already mentioned, we are interested only in increasing values for the histograms f_{kli} for the error E_G. Thus, we sum only over the positive steps from s to $s + 1$. With

$$S(y) = \begin{cases} y & \text{if } y > 0 \\ 0 & \text{else} \end{cases},$$

we can formulate

$$H_{kl}(i) = \sum_{s=1}^{h} \mathrm{S}(f_{kli}(s+1) - f_{kli}(s)) \tag{3}$$

Algorithm 2 Splitting of A_k

```
def Split($A_k$):
    $\hat{l} \leftarrow \underset{l}{\text{argmin}} \; \sum_{i=1}^{h} \sum_{s=1}^{h} f_{kli}(s)$
    $m \leftarrow \underset{i}{\text{argmin}} \; \sum_{s=1}^{h} f_{k\hat{l}i}(s)$
    $Z_0 \leftarrow \{\vec{x}_j | \; \text{bin}_{k\hat{l}}(\vec{x}_j) < H_{k\hat{l}}(m)\}$
    $Z_1 \leftarrow X_k \backslash Z_0$
return $\{\text{PCA}(1, Z_0), \text{PCA}(1, Z_1)\}$
```

and finally describe the generation error as the sum over all axes and all bins along each axis. Then, the resulting number E_G is a heuristic measure if the axes lie close or along areas of locally maximal density by quantifying the density-gradient from a line segment.

$$E_G = \frac{1}{N(X_k)} \sum_{l=1}^{d_k} \sum_{i=1}^{h} H_{kl}(i) \qquad (4)$$

The number of bins h is defined, such that there are three samples in each bin on average (assuming a uniform distribution)

$$h = \left\lfloor \left(\frac{N(X_k)}{3} \right)^{\frac{1}{d_k}} \right\rfloor .$$

We use this load-factor of three throughout all experiments in this manuscript, since our simulation results indicate that this is a good compromise between the resolution of the histogram and a robust gradient computation H_{kl} on the histograms.

E. Splitting of a local-PCA unit

Based on the generation error, we can decide whether to break a local PCA unit apart to yield a more precise approximation of the underlying distribution for the data X_k. However, we still have to find a suitable bi-partition.

Initially (See algorithm 1) we begin with a single axis $d_k = 1$, that basically represents the first principal component of a global PCA. But when is a line a good representation of a set of data points? In Figure 3 are some examples which we intuitively would tend to accept only (ii) and reject (i, iii and iv) as a good approximation. To put our intuition into mathematical expressions, we suggest splitting the data samples along an axis, where the density of points that projected onto the axis, is minimal.

III. Simulation Results

We test the suggested greedy iteration method for local-PCA on synthetic data in two dimensions and compared the two different candidates for the generation error E_G (equation 4) and $\tilde{E}_G$ (equation 2).

A. Noisy-"S", Square and Vortex

First, we concentrate on the generation error E_G (equation 4). Since no global optimization is done, the greedy approach can yield to different results, if the random initialization of the data-set is changed. However, some typical different

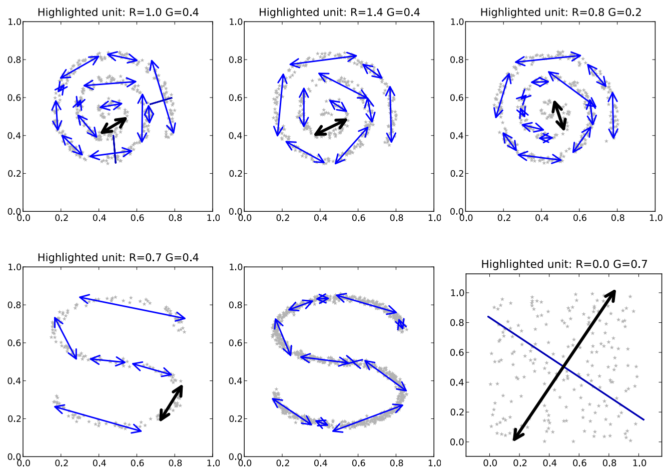

Fig. 4. **Top row: Different random initializations**. We draw 400 samples for three times from the same underlying probability density and show the final result of each run. Parameters are $t_R = 2.0$ and $t_G = 0.8$. One of the worst outcomes is shown to the left, and the right two plots show typical results.
Bottom row: Different synthetic data-distributions with the same parameter set. The two parameters are, again, set to $t_R = 2.0$ and $t_G = 0.8$. The left noisy-"S" consists of 200, the middle one of 1000 and the square has 200 data samples. Since the decision is done locally for each PCA-element, the algorithm can adapt (without changing the parameters) to different intrinsic dimensionalities of a manifold, here, one-dimensional and area-like manifolds. Each image shows the final-state, when the greedy iteration is finished. The reconstruction error for a two-dimensional distribution (right image) is zero. All plots in this figure are computed with the generation error E_G as defined in equation 4.

runs, as shown in the top row of figure 4, indicate that the overall result is promising, since the vortex-like distribution with only 400 samples is relatively sparsely populated. For example, Möller and Hoffmann[7] used 1000 sample points and 30000 iterations on this vortex-like data-set for a local-PCA method. We observe convergence typically for 15 to 20 steps.

The parameters t_G and t_R are – in a certain range – robust against the number of points and the intrinsic dimensionality. The result, in the bottom-row of figure 4 demonstrates that the intrinsic complexity and structure yields results, which are adapted to the manifold, without fine-tuning the parameters. The iterations shown in figure 1 and 4 use the same set of parameters.

B. Changing the thresholds t_G and t_R

To show the influence of the two parameters for the reconstruction and the generation error, we repeated the simulation with the noisy vortex of 400 samples. Figure 5 contains the plots for varying only t_R and figure 6 shows the results for simulations with different t_G. A lower threshold for the reconstruction error leads to an approximation of the manifold with more local PCA units. In the first two plots of figure 5, the highlighted line segments exhibit already a lower (local) reconstruction error than t_R and therefore some parts of the vortex are represented quite coarsely by a single element. The threshold t_G for the generation error (as defined in eq. 4), in contrast, does not look at the real

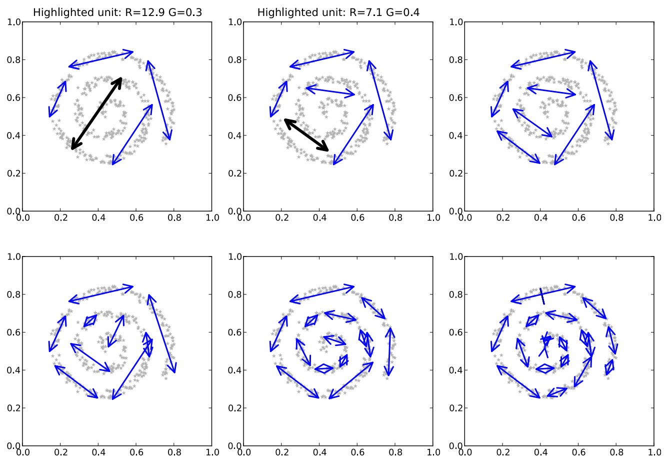

Fig. 5. **Influence of the threshold for the reconstruction error t_R.** Six different runs on the same 400 samples drawn from a vortex-like distribution. Row wise from top-left to bottom right with threshold values 15.0, 8.0, 5.0, 3.0, 1.0 and 0.5. The generation error E_G (equation 4) is kept fixed at 0.8 for each of these runs.

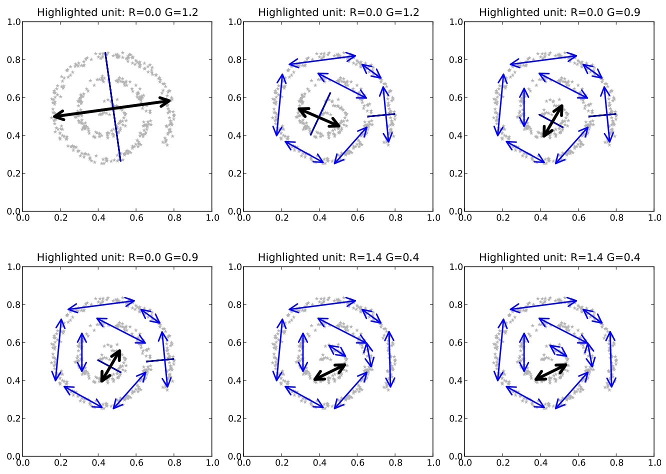

Fig. 6. **Influence of the threshold for the generation error t_G.** Row wise from top-left to bottom right with threshold values 1.3, 1.2, 1.1, 1.0, 0.9 and 0.8. The reconstruction error E_R is kept fixed at 2.0 for each of these runs. All six runs are performed on the same input samples. In the top-left plot, the distribution is seen as a 2d area and with lowering the target threshold, the inner windings are represented by more one-dimensional elements. The algorithm is not very sensitive to the exact value of t_G and in fact the final results for 1.1 and 1.0 are the same (as well as for 0.9 and 0.8).

distances from an axis, but only looks at the gradient of the density in the PCA-subspace. For one-dimensional PCA this is the density of the local set of points X_k, projected onto the principal axis. Therefore, the generation error E_G only measures how homogeneous the empirical distribution in the subspace is. For very high thresholds, the whole vortex is seen as a two-dimensional area (top-left plot of figure6). With lower thresholds, the inner area is represented in a finder resolution by more local PCA.

C. Comparison of the two alternatives $\tilde{E}_G$ and E_G

We compare the two candidates for the generation error as defined in equation 2 and 4 on a more complex synthetic data set, consisting of a ring and a filled square, connected by

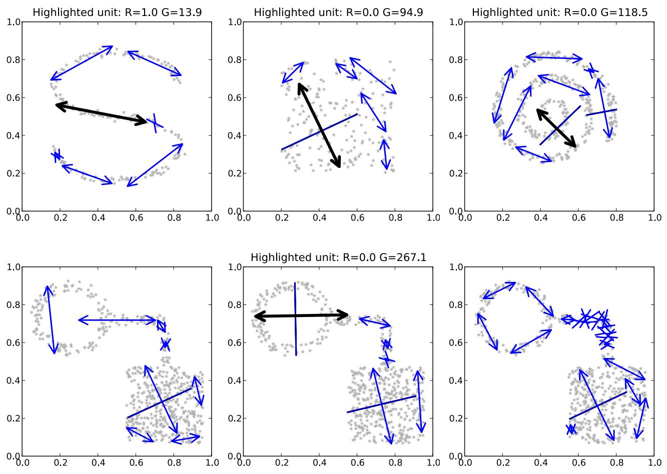

Fig. 7. **Results for the generation error, defined with the chi-square test.** Top-row: Again, the noisy-"S" and the uniform square with 200 samples and the vortex with 400 samples. The chi-square as a basis for the generation error $\tilde{E}_G$ has significant difficulties in the representation of larger areas of 2d-homogeneous density (in contrast to E_G, defined by the density gradient used in the other figures). The best threshold found for $\tilde{E}_G$ was 120 for these three examples. Higher values for the threshold t_G would yield to represent the square by a two-dimensional unit but hinder to detect the one-dimensional intrinsic dimension of the noisy-"S" and the vortex. Bottom-row: A more complex manifold. The three plots show different 600 random samples from the same underlying manifold. The threshold was set to 270 for these three different runs. Clearly the chi-square based generation error, is very sensitive to different random

a line. Figure 8 shows the iteration steps for the generation error E_G. One can observe that the homogeneous 2d area of the square is covered only by one two-dimensional PCA unit, whereas the line and the ring are approximated by several one-dimensional units. The fourth and fifth plot show the same step, but highlight different units and the generation error for the ring-like structure ($E_G = 1.3$) is significantly higher than for the square ($E_G = 0.7$). If we compare this to the generation error $\tilde{E}_G$ based on the chi-squared test (bottom row in figure 7), we find that the results for the latter $\tilde{E}_G$ are quite sensitive to different random initializations and perhaps more important, the filled square is approximated by several one-dimensional units in addition to a central two-dimensional one. Further, the ring-like substructure is not approximated very well.

IV. Conclusion

Our approach, in its greedy nature, may become a fast alternative to global optimization schemes, and, we are looking forward to evaluate the proposed method on higher-dimensional real-world data in the future. Due to the antagonistic behavior of the two error measures E_R and E_G and the way the greedy iteration is defined, the algorithm converges quickly. Another positive property is the robustness against the different intrinsic dimensionalities.

However, on the other hand, a split operation cannot be undone and this sometimes leads to unnecessary small units (compare Figure 4). A possibility to solve that problem would be a kind of "clean-up" pass at the end, in which one could try to merge small units together, of course only if the

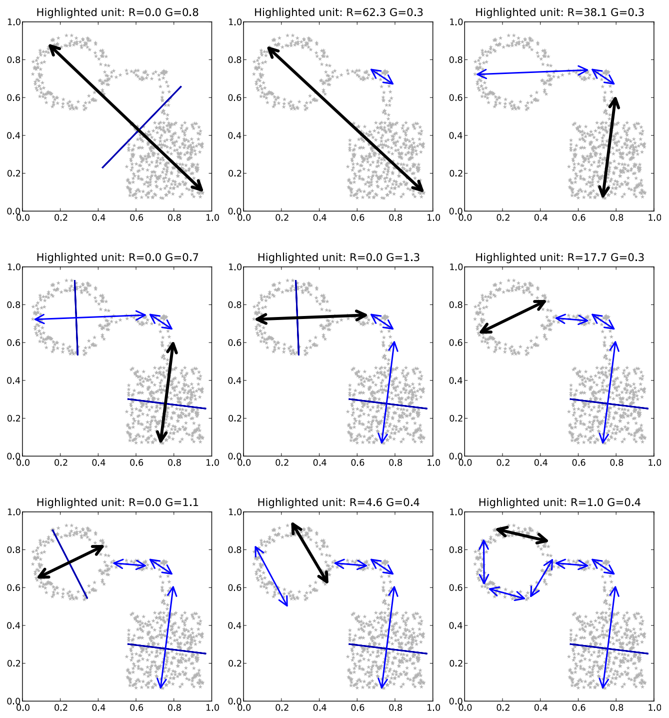

Fig. 8. **Iteration on a synthetic ring-line-square data set.** 600 sample points are randomly chosen. Depicted are all 8 iteration steps (there are two images of step four). The threshold for the generation error (E_G as in eq. 4) is 0.8 and t_R is 2.0.

merged units still are compatible to the given thresholds t_G and t_R.

On the one hand, we have demonstrated that our method can deal successfully with relatively few samples, on the other hand, the current algorithm is sensitive to outliers. The reason for this is that the PCA itself is known to be sensitive to outliers. Basically there are two possible solutions. First, to apply a known outlier-elimination technique and secondly, to prune away local PCA units that only represent very few points. Obviously this proposed extension can only be applied, if the idea of merging local PCA units is implemented, too. The reason for this is that units should only be pruned away, if they are “far off”. As mentioned before, sometimes there are relatively small units (compare figure 4), too, that are a result of the greedy splitting and one would prefer some of these to be combined with their neighbours.

Besides the before mentioned limitations and extensions, the main point is that for local PCA, it is not necessary to optimize for the positions, number of PCA units, dimensions and, last but not least, the partition of the input space. A good solution can be found with a fast iterative greedy approach.

References

[1] M. Brand. Charting a manifold. *Advances in Neural Information Processing Systems*, pages 985–992, 2003.

[2] S. Edelman and N. Intrator. Learning as extraction of low-dimensional representations. *Psychology of Learning and Motivation*, 36:353–380, 1997.

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

[4] Teuvo Kohonen. The self-organizing map. In *Proc. of the IEEE*, volume 78, pages 1464–1480, 1990.

[5] T. Martinetz, K. Schulten, et al. A "neural-gas" network learns topologies. *Artificial neural networks*, 1:397–402, 1991.

[6] P. Meinicke and H. Ritter. Local PCA learning with resolution-dependent mixtures of Gaussians. In *Artificial Neural Networks, 1999. ICANN99*, volume 1, pages 497–502, 1999.

[7] R. Möller and H. Hoffmann. An extension of neural gas to local PCA. *Neurocomputing*, 62:305–326, 2004.

[8] R. L. Plackett. Karl pearson and the chi-squared test. *International Statistical Review*, 51(1):59–72, 1983.

[9] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.