Texture Description by Independent Components

Dick de Ridder[1], Robert P. W. Duin[1], and Josef Kittler[2]

[1] Pattern Recognition Group
Dept. of Applied Physics, Faculty of Applied Sciences, Delft University of Technology
Lorentzweg 1, 2628 CJ Delft, The Netherlands
Phone +31 15 278 1845, Fax +31 15 278 6740
http://www.ph.tn.tudelft.nl/~dick
[2] Centre for Vision, Speech and Signal Processing
School of Electronics, Computing and Mathematics, University of Surrey
Guildford GU2 7XH Surrey, United Kingdom

Abstract. A model for probabilistic independent component subspace analysis is developed and applied to texture description. Experiments show it to perform comparably to a Gaussian model, and to be useful mainly for problems in which the detection of little occurring, high-frequency image elements is important.

1 Introduction

In many applications of statistical pattern recognition techniques to image data, images (or image patches) are described as high-dimensional vectors in which each element corresponds to a pixel position in the rectangular image grid. A problem associated with this approach is that estimating parameters in such a high-dimensional space is cumbersome. Furthermore, images that make sense to human observers only form a small subset of all possible positions in this space. The vector description of images therefore contains more parameters than necessary. Irrespective of these problems, one would often like to model image information locally invariant to offset, contrast, translation, rotation and scale. However, if an image patch is just slightly brightened, contrast enhanced, translated, rotated or scaled, the distribution of the vectors will change drastically, whereas to a human observer the image content looks very similar. That is, the representation is not naturally invariant.

It can be shown that transformed versions of an image patch all lie on an m-dimensional manifold in the d-dimensional space spanned by all pixel values, where m is the number of degrees of freedom present in the transformations [10]. Although this manifold may be intrinsically low-dimensional, it is likely to be nonlinear and lie folded up in the d-dimensional space. A good representation would therefore be one which describes this manifold using a small number of parameters, thereby avoiding the estimation problems in high-dimensional spaces and enforcing invariance to elementary transformations.

T. Caelli et al. (Eds.): SSPR&SPR 2002, LNCS 2396, pp. 587–596, 2002.

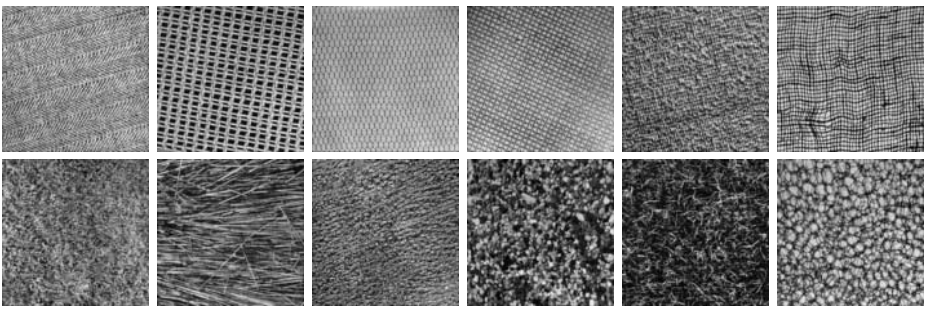


Fig. 1. The 12 Brodatz textures used in the experiments in this paper. Top: structured textures S1–S6; bottom: natural textures N1–N6

Earlier work [2,3,4] discussed the use of mixtures-of-PCA for locally modelling such manifolds, in applications to texture segmentation and image description. The main focus in this paper is on the use of independent component analysis (ICA) for describing sets of image patches. ICA is a recently proposed technique for finding subspaces in which the projected data distributions are independent. There have been reports of modelling sets of natural image patches using ICA (e.g. [6]), leading to sets of wavelet-like basisvectors. However, there have been no true applications of the subspaces thus found. In this paper, ICA will be applied as a texture description method and compared to a full Gaussian model.

The remainder of the paper is laid out thus: Sect. 2 discusses the texture image data used in the experiments. Next, in Sect. 3 the Gaussian and ICA models will be discussed. Sect. 4 describes experiments in texture description. Questions raised by these experiments with regard to ICA will be answered in Sect. 5, followed by some conclusions in Sect. 6.

2 Texture Data

For the experiments in this paper, two sets of textures were taken from the Brodatz album [1], shown in Fig. 1. It is to be expected that subspace methods will work better on the structured textures, as these exhibit stronger correlation between pixels. The original images were rescaled to make sure texel sizes (i.e. the sizes of the 'structuring elements") in the structured textures were smaller than 16×16 pixels (the image patch size used in the experiments). For each image, the grey value range was rescaled to $[0, 1]$.

As the models will be trained on image patches rather than entire images, training sets will have to be sampled from the images. To this end, 16×16 pixel windows were used. Furthermore, the notion of *episodes* introduced by Kohonen [8] was used. This artificial enlargement of the data set allows for incorporation of prior knowledge, i.e. that (subspace) models should be invariant to small amounts of translation, rotation and scaling. Next to a sample extracted at position (x_0, y_0), four translated samples are extracted at positions $(x_0 + x, y_0 + y)$,

where $x \sim U(-5,5)$ and $y \sim U(-5,5)$; five samples are taken at the same position in images rotated over -45°, -22.5°, 0°, 22.5°and 45°; and five samples are taken at the same position in images scaled to 1.1×, 1.2×, 1.3×, 1.4× and 1.5× the original size. This gives a total of 15 samples per episode. Data sets of 1,500 samples *per texture* were created in this way, containing 100 episodes.

The models used in this paper cannot be fitted when certain dimensions do not contain data, i.e. when there is zero variance. Therefore, directions in which there is little or no variance (assumed to contain noise) can be removed using PCA. This is called *pre-mapping*; the data is projected onto the eigenvectors corresponding to the set of largest eigenvalues explaining more than 90% of the variance. The data is not whitened. For 16×16 windows taken out of structured textures, this leaves 50 dimensions on average; out of natural textures, on average 70 dimensions remain.

3 Models

This section discusses two basic models that can be applied to texture description: the Gaussian model and the independent component analysis (ICA) model. The basis for the discussion will be the log-likelihood, which in segmentation algorithms would be used to assign image windows to models. A data set of d-dimensional vectors $\boldsymbol{x}$ will be denoted by $\mathcal{L} = \{\boldsymbol{x}^n\}, n = 1, \ldots, N$. For the ICA model, m will denote the number of independent components.

3.1 Gaussian

The log-likelihood of a sample $\boldsymbol{z}$ belonging to a Gaussian distribution is:

$$\mathfrak{L}_{\text{Gauss}}(\boldsymbol{z}|\boldsymbol{\mu}, \boldsymbol{C}) = -\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln|\det(\boldsymbol{C})| - \frac{1}{2}(\boldsymbol{z}-\boldsymbol{\mu})^T\boldsymbol{C}^{-1}(\boldsymbol{z}-\boldsymbol{\mu}) \quad (1)$$

where $\boldsymbol{\mu}$ is the mean and $\boldsymbol{C}$ the covariance matrix, estimated on all $\boldsymbol{x} \in \mathcal{L}$. For a number of texture data sets, the inverse of $\boldsymbol{C}$ cannot be calculated in a 256D space; hence PCA pre-mapping (see the previous section) was applied.

3.2 Independent Component Analysis

Lately, independent component analysis or ICA [7] has enjoyed considerable attention. In its most basic form, this method tries to find directions in which the data is independent (rather than just decorrelated as in principal component analysis, PCA). The linear ICA model is $\boldsymbol{u} = \boldsymbol{W}(\boldsymbol{x}-\boldsymbol{\mu})$, in which $\boldsymbol{u}$ are the m-dimensional projected vectors. As ICA was originally applied to blind separation of various signals (or *sources*) $\boldsymbol{s}$ under an additive model, the matrix $\boldsymbol{W}$ is often called the *unmixing matrix* and the backprojection matrix $\boldsymbol{A}$ the *mixing matrix*. This backprojection is defined as $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{\mu}$. Note that in the notation here, $\boldsymbol{u}$ is used to denote an estimate of the *sources* $\boldsymbol{s}$.

Several (related) iterative algorithms have been proposed to perform ICA. For this work, an algorithm was developed as a variation on the extended infomax ICA algorithm of Lee et al. [9]. Lee's algorithm is a maximum likelihood (ML) fit of a set of distributions, in which for each component u_i it is learned whether to fit a sub-Gaussian or a super-Gaussian distribution. The number of independent components m is assumed to be equal to the number of dimensions in the data d. The log-likelihood can be expressed as [9]

$$\mathfrak{L}_{\mathrm{ICA}}(\boldsymbol{z}|\boldsymbol{A}) = -\ln|\det(\boldsymbol{A})| + \ln p(\boldsymbol{u}) \tag{2}$$

where $\boldsymbol{u} = \boldsymbol{W}\boldsymbol{z}$ and $\boldsymbol{W} = \boldsymbol{A}^{-1}$, which is possible since $m = d$ and therefore $\boldsymbol{A}$ is a square matrix. For notational simplicity, $\boldsymbol{\mu}$ is assumed to be $\mathbf{0}$. An ICA base can be found using a generalised EM algorithm to maximise Eqn. 2, in which the estimate for $\boldsymbol{W}$ is updated by a gradient descent learning rule. This batch learning rule, for the entire dataset (where $\boldsymbol{U}$ is the matrix containing the source approximations $\boldsymbol{u}$ as its columns and $\boldsymbol{X}$ likewise contains the data vectors), is

$$\Delta\boldsymbol{W} = \eta\left[N\boldsymbol{A}^T + \phi(\boldsymbol{U})\boldsymbol{X}^T\right] \tag{3}$$

where η is a learning rate and $\phi_i(u_i) = \frac{\partial \ln p_i(u_i)}{u_i}$, with $p_i(u_i)$ a sub-Gaussian or super-Gaussian distribution:

$$p_i(u_i) = \begin{cases} f_{sub}\,(u_i) = \frac{1}{2}\left(f_{Gauss}(u_i;1,1) + f_{Gauss}(u_i;-1,1)\right) \\ f_{sup}\,(u_i) = c\, f_{Gauss}\,(u_i;0,1)\operatorname{sech}(u_i) \end{cases} \tag{4}$$

with $f_{Gauss}(u_i;\mu,\sigma^2)$ the Gaussian distribution function and c a normalisation constant. Rule 3 was (approximately) found by various authors (e.g. [**?**]); note that the assumptions made to find it were (a) that there is no noise and (b) that matrix $\boldsymbol{A}$ is full rank, i.e. there are as many sources m as there are mixtures d.

The main limitation of this ML algorithm is that it works only for $m = d$. In the case of texture description, it is not to be expected that there are that many ICs. Therefore, a new learning rule was derived for the *undercomplete* case where $m < d$, by modelling the remaining dimensions as Gaussian noise [2]. This violates the assumptions necessary for the learning rule derived above.

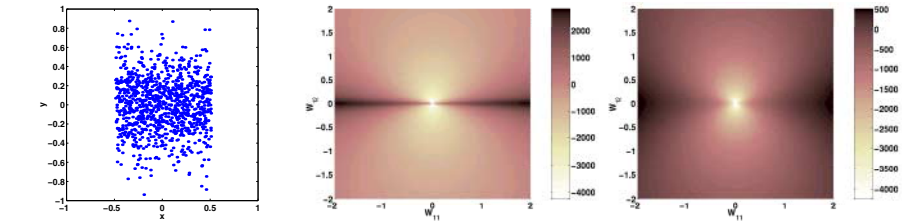
The revised model contains an added noise term, $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{\mu} + \boldsymbol{\epsilon}$, with $\boldsymbol{s}$ now an m-dimensional vector, $\boldsymbol{A}$ a $d \times m$ matrix and $\boldsymbol{\epsilon}$ Gaussian i.i.d. distributed noise with $\boldsymbol{C} = \sigma^2\boldsymbol{I} = \frac{1}{\beta}\boldsymbol{I}$. The log-likelihood now becomes

$$\mathfrak{L}_{\mathrm{ICA}}(\boldsymbol{z}|\boldsymbol{A},\beta) \approx \frac{d}{2}\ln\frac{\beta}{2\pi} - \frac{\beta}{2}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{u})^T(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{u}) - \frac{1}{2}\ln|\det(\boldsymbol{A}^T\boldsymbol{A})| + \ln p(\boldsymbol{u}) \tag{5}$$

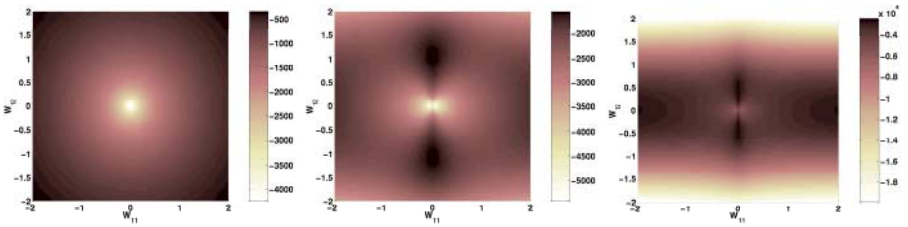
and the batch learning rule changes to

$$\Delta\boldsymbol{W} = \eta\left[\beta N\boldsymbol{A}^T\boldsymbol{C}(\boldsymbol{I} - \boldsymbol{A}\boldsymbol{W}) + N\boldsymbol{A}^T + \phi(\boldsymbol{U})\boldsymbol{X}^T\right] \tag{6}$$

The first term is an orthogonalisation term which works in the space rotated and scaled by $\boldsymbol{C}$. Note that in this model $\boldsymbol{W} = \boldsymbol{A}^{-1}$ no longer holds since $\boldsymbol{A}$ is not a square matrix; instead, $\boldsymbol{W} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T$.



(a) $\sigma^2 \approx 0.2872$ (b) $\sigma^2 = 0.01$ (c) $\sigma^2 = 0.10$



(d) $\sigma^2 = 0.2872$ (e) $\sigma^2 = 1.0$ (f) $\sigma^2 = 2.5$

Fig. 2. (a) A simple 2D data set. (b)-(f) $\mathfrak{L}_{\mathrm{ICA}}$ as a function of $\boldsymbol{W}$. For small σ^2, the local maxima (dark spots) correspond to the independent component (horizontal); for large σ^2, they correspond to the Gaussian component (vertical)

Eqn. 6 gives the basic learning rule for $\boldsymbol{W}$. However, the noise parameter β has to be estimated as well. This can easily be done using the GEM algorithm. In each step, estimate the current parameters in the E-step by:

$$\boldsymbol{u}^n = \boldsymbol{W}\boldsymbol{x}^n,\ n = 1, \ldots, N \quad (7) \qquad \boldsymbol{v}^n = \boldsymbol{W}^\top \boldsymbol{x}^n,\ n = 1, \ldots, N \quad (8)$$

$$\beta^{-1} = (d-m)^{-1} \sum_{i=1}^{d-m} \mathrm{var}(v_i) \quad (9) \qquad \boldsymbol{A} = \boldsymbol{W}^T (\boldsymbol{W}\boldsymbol{W}^T)^{-1} \quad (10)$$

where $\boldsymbol{W}^\top$ is the nullspace of $\boldsymbol{W}$, i.e. β corresponds to the inverse of the average noise outside the subspace (cf. probabilistic PCA [11]). In the M-step, the log-likelihood is then maximised by applying the learning rule (Eqn. 6) to $\boldsymbol{W}$.

The estimation of β has a large influence on convergence. It controls the trade-off between finding a subspace (the 2^{nd} term in Eqn. 5) and independent components (the 4^{th} term). This is illustrated in Fig. 2. The data set consists of 2D samples of which the x-coordinate is drawn from a $U(-0.5, 0.5)$ distribution with $\sigma^2 \approx 0.2872$ and the y-coordinate is drawn from a Gaussian distribution with $\mu = 0$ and variable σ^2. If β is fixed at σ^{-2}, the algorithm converges for any setting of σ. However, if β is learned, this is not always the case. For various settings of σ^2, Fig. 2 indicates $\mathfrak{L}_{\mathrm{ICA}}$ as a function of $\boldsymbol{W}$. Clearly, as σ increases, finding the independent component becomes less likely than finding the Gaussian component, depending on initialisation of $\boldsymbol{W}$.

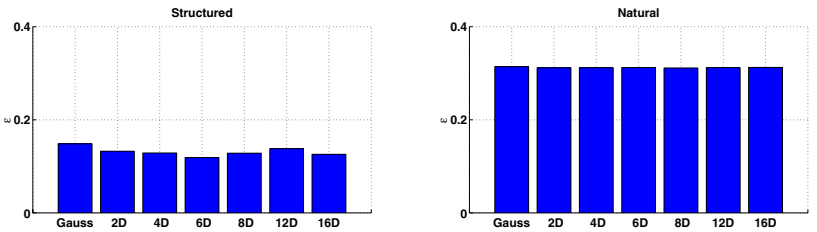


Fig. 3. Predicted error ε on texture sets, for the Gaussian model and ICA models of varying dimensionalities

This exposes a problem of most ICA algorithms: to find only independent components, the role variance plays will have to be eliminated. Although alternatives exist, the easiest method is by whitening the data before applying the ICA subspace algorithm, i.e. using $\boldsymbol{x}' = \boldsymbol{C}^{-\frac{1}{2}}\boldsymbol{x}$. This will make the data variance 1 in all directions. Pre-whitening also simplifies the algorithm in a number of ways: β can be fixed at 1; the rows of $\boldsymbol{W}$ and columns of $\boldsymbol{A}$ can be constrained to have unit length; and in the batch learning rule (Eqn. 6), the first term drops out, as $\boldsymbol{C} = \boldsymbol{I}$, so the learning rule is identical to the one originally proposed by Lee at al. (Eqn. 3). The ICA subspace model essentially changes to a Gaussian model in which m directions should contain non-Gaussian distributions.

4 Experiments

In a simple experiment, the Gaussian model and ICA were compared on their ability to describe texture[1]. Individual models M_i $(i = 1, \ldots, 6)$ were trained on the 6 individual Brodatz texture images, for both the structured and the natural texture set. After training, the negative log-likelihood of each pixel in each texture image I_j of belonging to each model M_i was calculated. This gives a set of distances F_{ij} between model M_i and texture image I_j. As these distances are approximately normally distributed[2], a simple performance measure is [5]:

$$\varepsilon = \underset{i=1,\ldots,6;\, j\neq i}{\text{med}} \frac{1}{2} - \frac{1}{2}\text{erf}\left(\frac{1}{2}\sqrt{\mathcal{F}(F_{ii}, F_{ij})}\right) \tag{11}$$

where $\mathcal{F}(F_1, F_2)$ is the Fisher distance between two distributions F_1 and F_2 and $\varepsilon \in [0, 0.5]$ is the median *predicted Bayes error*, assuming normally distributed features with equal variance and equal prior probabilities[3]. The median is taken over all models and all textures to prevent outliers from having too much influence.

[1] For details on experimental settings, see [2].

[2] The distribution of a likelihood calculated in d dimensions can be approximated by a χ^2_d distribution, which for large d can be approximated by a normal distribution.

[3] Note that the per-model pre-mapping makes this measure pessimistic; in a true segmentation application, PCA pre-mapping would be performed on the entire data set rather than single textures. However, relative performance is not influenced.

The results (Fig. 3) show, firstly, that the predicted error indeed is pessimistic; in [2], the Gaussian model is applied in a mixture model setting and gives very good segmentation results. However, of interest for this paper is the fact that the Gaussian and ICA models give nearly equal performance. Furthermore, for an increasing number of independent components performance hardly changes. This raises the question as to what exactly the advantage of modelling dimensions by non-Gaussian distributions is.

5 Independent Component Analysis

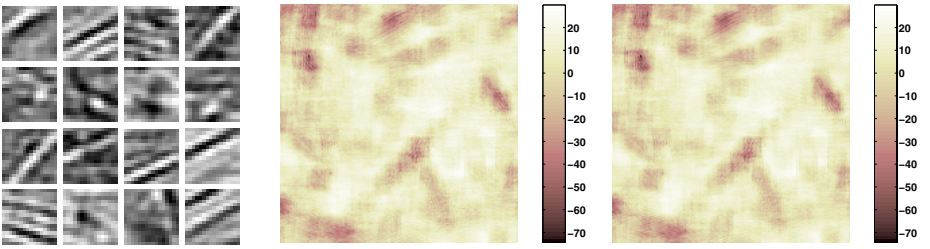
Investigation of the kurtoses of the distributions of data projected onto the extracted independent components shows that non-Gaussian directions have indeed been found: the kurtoses of the projected data distributions deviated significantly from 3 for any model. The algorithm works.

If the ICA model really fits better than the Gaussian, its likelihood $\mathfrak{L}_{\text{ICA}}$ (Eqn. 5) should be significantly larger than that of the Gaussian, $\mathfrak{L}_{\text{Gauss}}$. However, for nearly all textures, the average increase in likelihood (over all pixels) was negligible. To get a better idea of why these increases are so low, the "straw" natural texture N2 (Fig. 1) is considered. Fig. 4 (a) shows the ICA basis vectors found after training a 16D ICA model. Clearly, they correspond to directed edge detectors, which one would expect to be of use in segmenting the straw image. To see their effect, for each pixel in the original image the likelihood of the window of which it is the center can be plotted, again as an image. Figs. 4 (b) and (c) show these likelihood images for both the 64D ICA model and the Gaussian model. At first glance, there is no difference between the two. However, there is a difference, albeit small; Fig. 4 (d)-(i) shows this difference for an increasing number of independent components in the ICA model. For presentation purposes, negative differences (which fell in the range $[-2, 0]$) have not been shown.

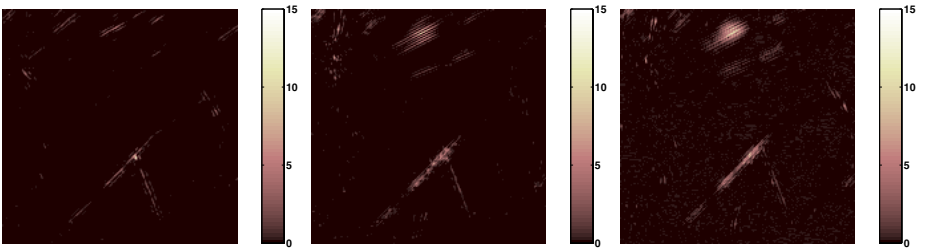
It now becomes obvious why the ICA model shows no improvement over the Gaussian model. In general, the independent components correspond to characteristic but more or less unique high-frequency events in images. For this image, these are the few straws that have a different orientation than the majority. The first few independent components (2D-4D models) correspond to the straws below the center of the image; as more dimensions are added, the straws at the top of the image get modelled better as well. Finally, as more and more independent components are found, all straws with non-standard orientations are singled out (relative to the Gaussian model), whereas the main structure of the texture becomes slightly more likely, but in a noise-like fashion. In fact, this happens for all textures.

6 Conclusions

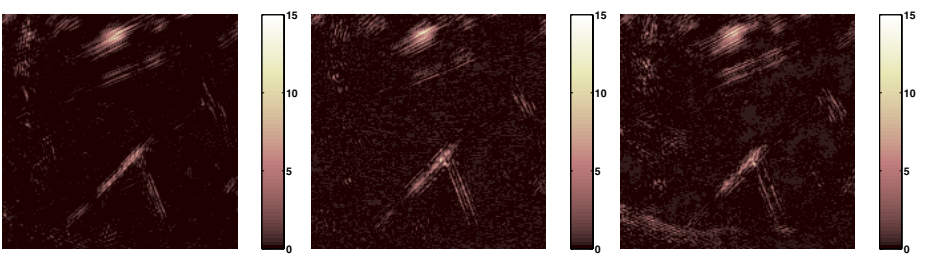
A probabilistic subspace ICA model was developed and compared to a Gaussian model on the task of texture description. It was shown that data pre-whitening is necessary to be able to train the ICA model, which effectively makes the ICA



(a) $\boldsymbol{A}$, 16D ICA (b) $\mathfrak{L}_{\mathrm{ICA}}$, 64D (c) $\mathfrak{L}_{\mathrm{Gauss}}$



(d) $\mathfrak{L}_{\mathrm{ICA}} - \mathfrak{L}_{\mathrm{Gauss}}$, 2D (e) $\mathfrak{L}_{\mathrm{ICA}} - \mathfrak{L}_{\mathrm{Gauss}}$, 4D (f) $\mathfrak{L}_{\mathrm{ICA}} - \mathfrak{L}_{\mathrm{Gauss}}$, 6D



(g) $\mathfrak{L}_{\mathrm{ICA}} - \mathfrak{L}_{\mathrm{Gauss}}$, 8D (h) $\mathfrak{L}_{\mathrm{ICA}} - \mathfrak{L}_{\mathrm{Gauss}}$, 12D (i) $\mathfrak{L}_{\mathrm{ICA}} - \mathfrak{L}_{\mathrm{Gauss}}$, 16D

Fig. 4. (a) ICA basis vectors found on texture N2; (b) log-likelihood of each pixel belonging to a 64D ICA model; (c) same for a Gaussian model and (d)-(i) the difference between $\mathfrak{L}_{\mathrm{ICA}}$ of various dimensionalities and $\mathfrak{L}_{\mathrm{Gauss}}$

model a Gaussian one in which a subset of dimensions have non-Gaussian distributions. Subsequently, both models were shown to perform equally well on texture description. Experimental observations led to the conclusion that: (a) the increase in likelihood is generally so low, that the Gaussian model may be considered as good a model as ICA for a data set consisting of texture image patches; (b) ICA – given its high computational complexity – is not useful as a

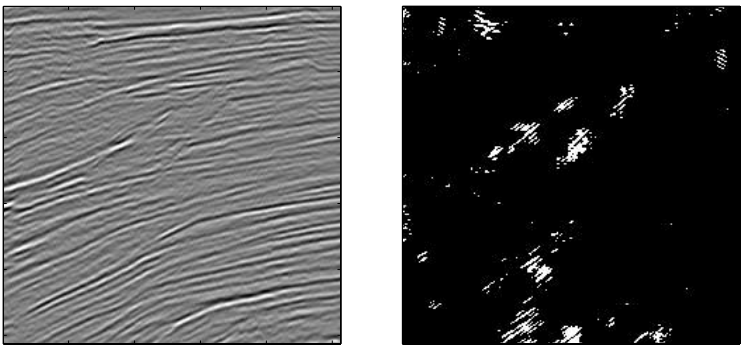


Fig. 5. A useful application of ICA: left, a seismic image; right, thresholded $(\mathfrak{L}_{\mathrm{ICA}} - \mathfrak{L}_{\mathrm{Gauss}})$, for $m = 8$

texture description tool. For segmentation, the goal is to find models which describe textures in a shift-invariant way. To this end, non-standard regions should be *ignored* rather than modelled. The Gaussian model does this by modelling the data by (co)variance only, ignoring outliers. Under the ICA model, as it focuses on non-Gaussianity alone, outliers are more probable (for super-Gaussian sources); in fact, the sources are indirectly optimised to make outliers more likely.

The main conclusion is that, where many authors have suggested ICA might be useful for image processing, its application area is limited to problems in which the detection of unique, characteristic events is of importance. As an example, consider the seismic image shown in Fig. 5. The Gaussian model will pick up only on the two dominant directions present. However, the ICA model is better at description of some faint – yet to a human observer clearly visible – discontinuities and bifurcations in the image.

Acknowledgements

This work was partly supported by the Foundation for Computer Science Research in the Netherlands (SION), the Dutch Organisation for Scientific Research (NWO), the Foundation for Applied Sciences (STW) and the Engineering and Physical Sciences Research Council (EPSRC) in the UK (grant numbers GR/M90665 and GR/L61095).

References

1. P. Brodatz. *Textures, a photographic album for artists and designers*. Dover Publications, New York, NY, 1966. 588
2. D. de Ridder. *Adaptive methods of image processing*. PhD thesis, Delft University of Technology, Delft, 2001. 588, 590, 592, 593
3. D. de Ridder, J. Kittler, O. Lemmers, and R.P.W. Duin. The adaptive subspace map for texture segmentation. In *Proc. ICPR 2000*, pages 216–220, Los Alamitos, CA, 2000. IAPR, IEEE Computer Society Press. 588

4. D. de Ridder, O. Lemmers, R.P.W. Duin, and J. Kittler. The adaptive subspace map for image description and image database retrieval. In *Proc. S+SSPR 2000*, pages 94–103, Berlin, 2000. IAPR, Springer-Verlag. 588
5. K. Fukunaga. *Introduction to statistical pattern recognition.* Electrical Science Series. Academic Press, NY, NY, 1972. 592
6. J. Hurri. Independent component analysis of image data. Master's thesis, Dept. of Computer Science and Engineering, Helsinki University of Technology, Espoo, Finland, March 1997. 588
7. A. Hyvärinen. Survey on independent component analysis. *Neural Computing Surveys*, 1(2):94–128, 1999. 589
8. T. Kohonen, S. Kaski, and H. Lappalainen. Self-organized formation of various invariant-feature filters in the Adaptive-Subspace SOM. *Neural Computation*, 9(6):1321–1344, 1997. 588
9. T.-W. Lee, M. Girolami, and T.J. Sejnowski. Independent component analysis using an extended infomax algorithm for mixed sub-Gaussian and super-Gaussian sources. *Neural Computation*, 11(2):417–441, 1999. 590
10. H. Lu, Y. Fainman, and R. Hecht-Nielsen. Image manifolds. In *Applications of Artificial Neural Networks in Image Processing III, Proceedings of SPIE*, volume 3307, pages 52–63, Bellingham, WA, 1998. SPIE, SPIE. 587
11. M.E. Tipping and C.M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999. 591