Fixed point rules for heteroscedastic Gaussian kernel-based topographic map formation

Marc M. Van Hulle
K.U.Leuven, Laboratorium voor Neuro- en Psychofysiologie
Campus Gasthuisberg, Herestraat 49, BE-3000 Leuven, BELGIUM
E-mail: marc@neuro.kuleuven.be

***Abstract*— We develop a number of fixed point rules for training homogeneous, heteroscedastic but otherwise radially-symmetric Gaussian kernel-based topographic maps. We extend the batch map algorithm to the heteroscedastic case and introduce two candidates of fixed point rules for which the end-states, *i.e.*, after the neighborhood range has vanished, are identical to the maximum likelihood Gaussian mixture modeling case. We compare their performance for clustering a number of real world data sets.

1 Introduction

Several topographic map formation algorithms have been introduced that employ Gaussian activation kernels rather than Voronoi regions, such as in the case of the popular Self-Organizing Map (SOM) algorithm and its adapted versions [5]. Reviews of the homoscedastic (equal-variance) Gaussian case have been introduced by Obermayer's group [3] and by Heskes [4]. Obermayer and coworkers showed the connection between different classes of Gaussian kernel-based topographic map formation algorithms and, as a limiting case, the batch map version of the SOM algorithm [5]. Heskes showed the connection between minimum distortion topographic map formation and maximum likelihood homoscedastic Gaussian mixture density modeling (GMM). A unifying account of the heteroscedastic case was introduced in [10], where the link was demonstrated between distortion minimization, log-likelihood maximization and Kullback-Leibler divergence minimization. Furthermore, kernels other than Gaussians have also been suggested such as the incomplete gamma- [9] and the Edgeworth-expanded Gaussian kernels [11]. Another idea for Gaussian kernel-based topographic map formation is to minimize the Kullback-Leibler divergence, as suggested by [1], using homoscedastic Gaussians, and extended by Yin and Allinson [12] to the heteroscedastic case.

The heteroscedastic kernels are expected to provide a better density estimate than their homoscedastic counterparts, and thus a better indication for the existence of clusters in the data, but the drawback of at least the heteroscedastic Gaussian- and heteroscedastic Edgeworth-expanded Gaussian cases is that no complete set of fixed point update rules can be derived from their objective functions, which can be a distortion-, energy- or a log-likelihood function [10, 11]. Only for the Gaussian kernel centers such a fixed point rule can be derived.

In this article, we first extend the traditional batch map algorithm to the heteroscedastic case. We then consider two candidates of fixed point rules for heteroscedastic (but homogeneous and radially-symmetric) Gaussian kernel-based topographic map formation for which the end-states, *i.e.*, after the neighborhood range has vanished, are identical to the maximum likelihood Gaussian mixture modeling case. We compare their topographic map formation behavior, and assess their performance for clustering a number of real world data sets. We also consider the original batch map in the comparison where possible.

2 Fixed point rules

2.1 Gaussian mixture modeling

As a starting point, we consider the case of homogeneous (equal prior probabilities), heteroscedastic (differing variances) Gaussian mixture density estimation:

$$p(\mathbf{v}) \approx \frac{1}{N}\sum_i K_i(\mathbf{v}, \mathbf{w}_i, \sigma_i), \tag{1}$$

with N the number of Gaussian kernels:

$$K_i(\mathbf{v}, \mathbf{w}_i, \sigma_i) = \frac{1}{(2\pi\sigma_i^2)^{\frac{d}{2}}} \exp\left(\frac{-\|\mathbf{v}-\mathbf{w}_i\|^2}{2\sigma_i^2}\right), \tag{2}$$

with center $\mathbf{w}_i = [w_{i1}, \ldots, w_{id}] \in \mathbb{R}^d$, and radius σ_i, and $\mathbf{v} = [v_1, \ldots, v_d]$ a random vector in $\mathbb{R}^d$ generated from the probability density $p(\mathbf{v})$.

A standard procedure to estimate the parameters $\mathbf{w}_i$ and σ_i, $\forall i$, is by maximizing the (average) likelihood, or by minimizing the (average) negative log-likelihood for the sample $\mathcal{S} = \{\mathbf{v}^\mu | \mu = 1, \ldots, M\}$ [7]:

$$F = -\log \mathcal{L} = -\frac{1}{M}\sum_\mu \log \frac{1}{N}\sum_i K_i(\mathbf{v}^\mu, \mathbf{w}_i, \sigma_i) \tag{3}$$

through an Expectation-Maximization (EM) approach [2]. This results in the following fixed point rules:

$$\mathbf{w}_i = \frac{\sum_\mu P(i|\mathbf{v}^\mu)\mathbf{v}^\mu}{\sum_\mu P(i|\mathbf{v}^\mu)},$$

Proceedings of the 6th International Workshop on Self-Organizing Maps (WSOM 2007)
Published by the Neuroinformatics Group, Bielefeld University, Germany, ISBN 978-3-00-022473-7
All contributions to WSOM 2007 are available online at: http://biecoll.ub.uni-bielefeld.de

$$\sigma_i^2 = \frac{\sum_\mu P(i|\mathbf{v}^\mu)\|\mathbf{v}-\mathbf{w}_i\|^2/d}{\sum_\mu P(i|\mathbf{v}^\mu)}, \quad (4)$$

where we have substituted for the posterior probabilities $P(i|\mathbf{v}^\mu) = \frac{K_i^\mu}{\sum_j K_j^\mu}$. Note that in the Expectation step, we update the posterior probabilities prior to the Maximization step, which is the update of the kernel centers and radii. Note also that we can easily extend the current format to non-homogeneous GMMs by considering the prior as an additional parameter.

As a reference example, we consider the two-dimensional uniform distribution $[-1,1]^2$ from which we take 1000 samples. We consider a mixture of $N = 25$ formal neurons of which we initialize the kernel centers by taking samples from the distribution; the kernel radii are initialized $\sigma_i = 0.2$, $\forall i$. We train the GMM during $t_{\max} = 100$ epochs. At $t_{\max}$, the average log-likelihood is -1.440 and the average kernel radius is 0.147 (see Fig. 2A,B stippled lines). The resulting distribution of the kernels is shown in Fig. 1A.

2.2 Batch map and its extension

The original batch map [5], called here BMo, is defined as follows:

$$\mathbf{w}_i = \frac{\sum_\mu \Lambda(i^*,i)\mathbf{v}^\mu}{\sum_\mu \Lambda(i^*,i)}, \quad (5)$$

with $i^* = \text{argmin}_i\|\mathbf{v}-\mathbf{w}_i\|$ and $\Lambda(i^*,i)$ the traditional neighborhood function (with arguments in lattice coordinates). Note that the definition of the "winner" neuron i^* is actually equivalent to looking for the homoscedastic Gaussian kernel with the largest activity, $i^* = \text{argmax}_i K_i$. Bearing this in mind, and observing that the neighborhood function can be regarded as a probability in lattice space (noise decoding [6]; confusion probability [4]), we can extend this rule to the heteroscedastic case:

$$\begin{aligned}\mathbf{w}_i &= \frac{\sum_\mu \Lambda(i^*,i)\mathbf{v}^\mu}{\sum_\mu \Lambda(i^*,i)},\\ \sigma_i^2 &= \frac{\sum_\mu \Lambda(i^*,i)\|\mathbf{v}-\mathbf{w}_i\|^2/d}{\sum_\mu \Lambda(i^*,i)},\end{aligned} \quad (6)$$

with $i^* = \text{argmax}_i K_i$ (which is no longer equivalent to $i^* = \text{argmin}_i\|\mathbf{v}-\mathbf{w}_i\|$, but which is required since we now have heteroscedastic kernels). We call this new EM algorithm the extended batch map (BMe). However, by the definition of the winner, the tails of the kernels are cut off, since the kernels overlap, so that these data points can not lead to a proper modeling of the kernel radii. As a result, the actual kernel radii will be underestimated.

To continue with our example, we consider a 5×5 lattice of which the kernels are initialized as above, and that we train with a Gaussian neighborhood function of which the range is decreased exponentially $\sigma_\lambda(t) = \sigma_\lambda(0)\exp(-2t/t_{\max})$; the initial range $\sigma_\lambda(0) = 2.5$. The average log-likelihood is shown in Fig. 2A,B (dotted lines). We observe that the radii quickly grow from their initial 0.2 value to over 0.7 and then converge to a value that is smaller than what is expected for the maximum likelihood approach (the average radius is now 0.111). The lattice at $t_{\max}$ is shown in Fig. 1B.

2.3 Proposition 1

The fixed point weight update rule which follows from minimum distortion topographic map formation [4] and which was also proposed in the Soft Topographic Vector Quantization (STVQ) algorithm [3], both for the homoscedastic case, is:

$$\mathbf{w}_i = \frac{\sum_\mu \sum_j \Lambda(j,i)P(j|\mathbf{v}^\mu)\mathbf{v}^\mu}{\sum_\mu \sum_j \Lambda(j,i)P(j|\mathbf{v}^\mu)}. \quad (7)$$

By observing the similar structure of the latter with the maximum likelihood approach, we propose the following set of fixed point rules for the heteroscedastic case:

$$\begin{aligned}\mathbf{w}_i &= \frac{\sum_\mu \sum_j \Lambda(j,i)P(j|\mathbf{v}^\mu)\mathbf{v}^\mu}{\sum_\mu \sum_j \Lambda(j,i)P(j|\mathbf{v}^\mu)},\\ \sigma_i^2 &= \frac{\sum_\mu \sum_j \Lambda(j,i)P(j|\mathbf{v}^\mu)\|\mathbf{v}^\mu-\mathbf{w}_i\|^2/d}{\sum_\mu \sum_j \Lambda(j,i)P(j|\mathbf{v}^\mu)}.\end{aligned} \quad (8)$$

We call this EM algorithm proposition 1 (Prop1).

We again consider the uniform distribution example with the same lattice and neighborhood settings as in the previous case. We now observe that this approach leads to rapidly growing kernel radii (dashed line in Fig. 2B), which causes the kernels to map the whole input space instead of the kernel centers, which stay at the centroid of the distribution (Fig. 1C). This is clearly a non-optimal maximum likelihood result (the dashed line in Fig. 2A is lower than the GMM result). However, this does not necessarily mean that, *e.g.*, for multimodal input densities, the kernel radii would also cover the whole space, as we will see later (see Fig. 6). One way to solve this problem is to smooth the radii updates over time, using a leaky integrator such as Wegstein's, $\sigma_i(t) \leftarrow (1-\alpha)\sigma_i(t-1) + \alpha\sigma_i(t)$ [5].

2.4 Proposition 2

In the heteroscedastic case that we proposed in [10], we applied a "bias-variance" decomposition of the weighted and normalized error term:

$$\sum_r \frac{\Lambda_{ri}}{2\sigma_r^2}\|\mathbf{v}^\mu-\mathbf{w}_r\|^2 = \frac{\|\mathbf{v}^\mu-\overline{\mathbf{w}}_i\|^2}{2\overline{\sigma}_i^2} + \sum_r \frac{\Lambda_{ri}}{2\sigma_r^2}\|\overline{\mathbf{w}}_i-\mathbf{w}_r\|^2, \quad (9)$$

with:

$$\overline{\mathbf{w}}_i = \frac{\sum_r \frac{\Lambda_{ri}\mathbf{w}_r}{\sigma_r^2}}{\sum_r \frac{\Lambda_{ri}}{\sigma_r^2}}, \quad (10)$$

$$\frac{1}{\overline{\sigma}_i^2} = \sum_r \frac{\Lambda_{ri}}{\sigma_r^2}. \quad (11)$$

Hence, we take as the ith kernel radius $\overline{\sigma}_i$ which we estimate from the σ_js in eq. (11). The σ_js are updated as in eq. (8). In this way, since we perform this weighted sum of inverse variances, we avoid the initially large radii generated by eq. (8). We call this EM algorithm proposition 2 (Prop2).

The EM algorithm for our uniform density now behaves better (full lines in Fig. 2A,B): both the average log-likelihood and the kernel radius are close to those of the maximum likelihood approach. The lattice is shown in Fig. 1D).

2.5 Other versions

One could also consider the format suggested in [10]:

$$\mathbf{w}_i = \frac{\sum_\mu \sum_j \Lambda(j,i) P(j|\mathbf{v}^\mu)\mathbf{v}^\mu}{\sum_\mu \sum_j \Lambda(j,i) P(j|\mathbf{v}^\mu)},$$

$$\sum_\mu \sum_j P(j|\mathbf{v}^\mu)(\overline{\sigma}_j)^2 \Lambda_{ij} =$$

$$\sum_\mu \frac{1}{d} \sum_j P(j|\mathbf{v}^\mu)\Lambda_{ij}\|\mathbf{v}^\mu - \mathbf{w}_i\|^2, \quad (12)$$

with $\overline{\sigma}_i$ as in eq. (11), which was derived from an objective function (distortion minimization, log-likelihood maximization), but which does not lead to a closed form solution for updating the kernel radii. This leads to a complex iterative update scheme, since for large neighborhood ranges, the determinant becomes close to singular, and since one should guarantee non-negative solutions for the σ_is. We do not further consider this update scheme.

A heuristic version is generated by adopting a mixed strategy, namely, to update the kernel radii as in maximum likelihood Gaussian mixture modeling, but to update the kernel centers with the neighborhood function present:

$$\mathbf{w}_i = \frac{\sum_\mu \sum_j \Lambda(j,i) P(j|\mathbf{v}^\mu)\mathbf{v}^\mu}{\sum_\mu \sum_j \Lambda(j,i) P(j|\mathbf{v}^\mu)},$$

$$\sigma_i^2 = \frac{\sum_\mu P(i|\mathbf{v}^\mu)\|\mathbf{v} - \mathbf{w}_i\|^2/d}{\sum_\mu P(i|\mathbf{v}^\mu)}. \quad (13)$$

However, it also leads to rapidly increasing kernel radii, and thus behaves similarly to Prop1, but, in addition, since it does not use neighborhood information for its kernel radii, it is more prone to getting trapped in solutions that do not show the correct the number of data clusters (as was observed with the data sets discussed next; results not shown). Therefore, it is not further considered.

3 Finally, there is the heuristic approach suggested by Yin and Allinson [12], which is minimizing the Kullback-Leibler divergence. Albeit that these authors only suggested an incremental, gradient-based learning procedure (thus, with a learning rate), we can easily cast their format into a fixed point learning scheme:

$$\mathbf{w}_i = \frac{\sum_\mu \Lambda(i^*,i) P(i|\mathbf{v}^\mu)\mathbf{v}^\mu}{\sum_\mu \Lambda(i^*,i) P(i|\mathbf{v}^\mu)},$$

$$\sigma_i^2 = \frac{\sum_\mu \Lambda(i^*,i) P(i|\mathbf{v}^\mu)\|\mathbf{v}^\mu - \mathbf{w}_i\|^2/d}{\sum_\mu \Lambda(i^*,i) P(i|\mathbf{v}^\mu)}, \quad (14)$$

with the winner neuron defined as $i^* = \mathrm{argmax}_i P(i|\mathbf{v}^\mu)$, thus, the neuron with the largest posterior probability. However, this learning scheme has very limited lattice unfolding capacity: *e.g.*, the lattice does not unfold for our uniform distribution example (Fig. 3A). There seems to be a confusion between the posterior probability and the neighborhood function (since their products are taken for the same kernel i): omitting the posteriors leads to the extended batch map algorithm, which is able to unfold the lattice. The neighborhood functions in the Prop1 and Prop2 EM rules act as smoothing kernels (summing over all posteriors), and do not impede lattice unfolding. To remedy the problem with Yin and Allinson's, one could replace $P(i|\mathbf{v}^\mu)$ by $P(i^*|\mathbf{v}^\mu)$ in the above equations. The lattice unfolds, but the solution is now similar to that of BMe, also with the underestimated kernel radius size (Fig. 3B). Hence, we do not further consider the Yin and Allinson approach.

3 Simulations

We now consider the following real-world data sets, which are all available from the UCI Machine Learning Repository (`http://www1.ics.uci.edu/~mlearn/MLRepository.html`): Iris ($M = 150$, $d = 4$), Wisconsin breast cancer ($M = 699$, $d = 9$), and wine recognition ($M = 178$, $d = 13$). The data set sizes M and dimensionalities d are listed between brackets. The Iris data set contains 3 classes (3 types of iris plants), the breast cancer set 2 classes (benign and malignant), and the wine recognition set also 3 classes (3 different wine cultivators).

In order to visualize the density surfaces in these higher dimensional spaces, and to inspect them for the presence of clusters, which are indicative of the different classes, we compute the densities at the kernel centers and at the doubly linearly interpolated positions between them (Fig. 4). The result is that a square lattice of N vertices becomes a square lattice of $16N - 24\sqrt{N} + 9$ vertices. Here we take a 5×5 lattice so that the interpolated lattice has 289 vertices. For all three examples, we normalize the data sets by subtracting the mean and by rescaling the dimensions so that they have unit variances. We initialize the kernel centers by

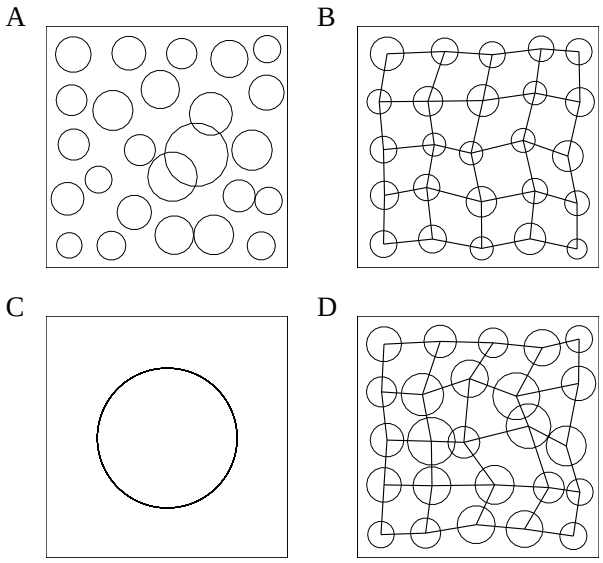

Figure 1: (A) Fixed point solution for a GMM with 25 kernels and for 1000 samples taken from the two-dimensional uniform distribution $[-1, 1]^2$ (boxes). The circles correspond to the standard deviation of the Gaussian kernels ("radii"). (B-D) Fixed point solutions for a 5×5 lattice of Gaussian kernels using the extended batch map BMe (B), and two update rules that correspond to the GMM fixed point rules, when the neighborhood has vanished, Prop1 (C) and Prop2 (D).

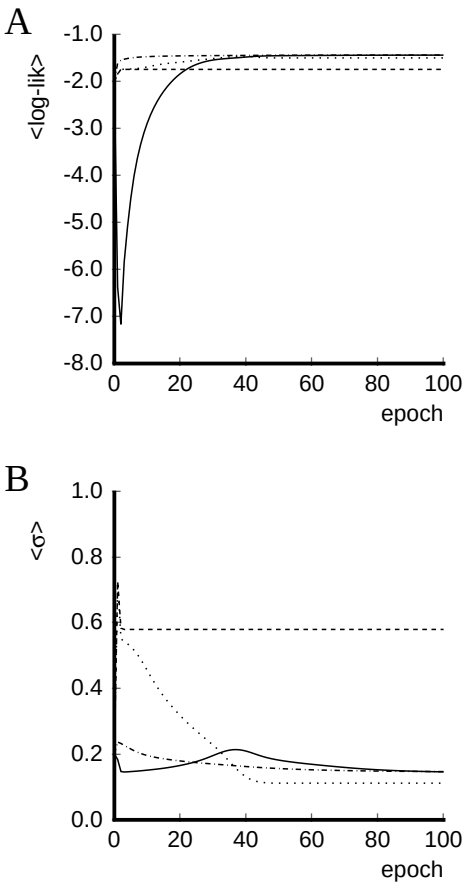

Figure 2: Average log-likelihood (A) and average kernel radius (B) as a function of training epochs for the update rules shown in Fig. 1, GMM (stippled line), BMe (dotted line), Prop1 (dashed line), and Prop2 (full line).

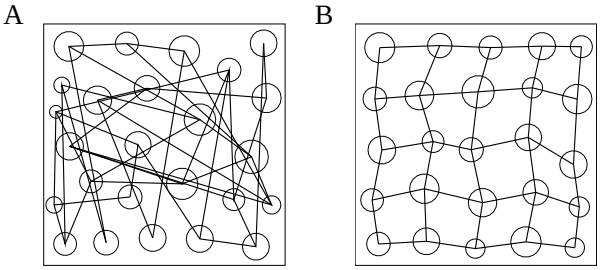

Figure 3: (A) Fixed point solution corresponding to Yin and Allinson's approach eq. (14). The lattice did not unfold. (B) Idem but now when replacing the posterior probability $P(i|\mathbf{v}^\mu)$ by $P(i^*|\mathbf{v}^\mu)$. The lattice is now unfolded.

Proceedings of the 6th International Workshop on Self-Organizing Maps (WSOM 2007)
Published by the Neuroinformatics Group, Bielefeld University, Germany, ISBN 978-3-00-022473-7
All contributions to WSOM 2007 are available online at: http://biecoll.ub.uni-bielefeld.de

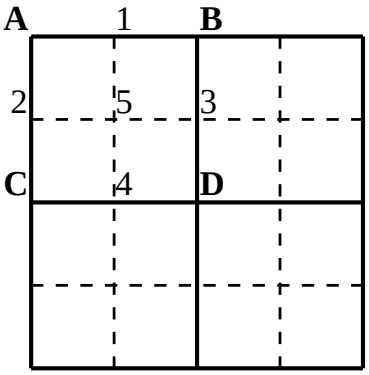

Figure 4: Computing the density map. The original lattice is shown in full lines; the interpolated lattice in dashed lines. For one quadrilateral in the original lattice, the interpolation procedure is detailed. The position of lattice point 1 in the input space is determined as the average of the kernel centers of lattice points A and B. The same holds for interpolated lattice points 2, 3 and 4. The input position of lattice point 5 is the average the input positions of lattice points 1 to 4. As a result, the linear lattice size L is doubled (actually $2L-1$). In order to double the lattice size again, this procedure is repeated by considering the original kernel centers and the interpolated points as the starting point of another interpolation run.

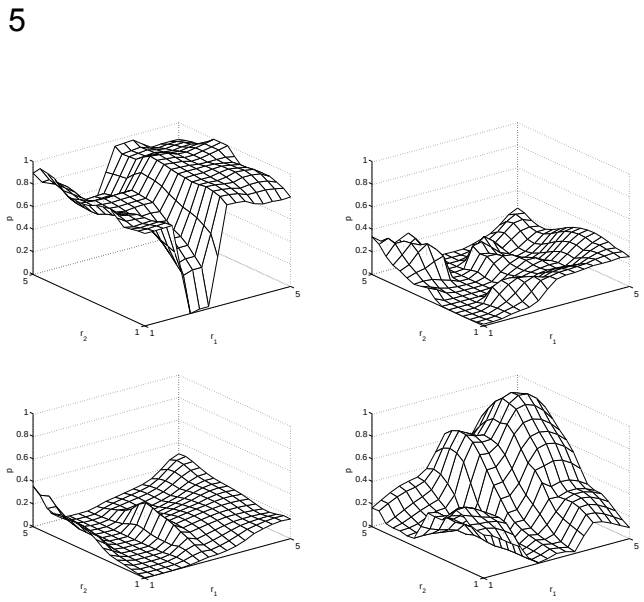

Figure 5: Results for the Iris data set. *Left upper panel:* U-matrix for the original batch map BMo obtained for a 5×5 lattice. *Right upper panel:* Density map for the extended batch map BMe. *Left lower panel:* Density map for Prop1. *Right lower panel:* Density map for Prop2. See text.

taking a uniform lattice in the square $[-1,1]^2$ in the plane formed by the first 2 principal components. The evolutions of the kernel radii are shown in Fig. 6.

The density maps using BMe, Prop1, and Prop2, and the one's complement of the U-matrix [8] for the original batch map (BMo), are shown in Fig. 5,7,8. For the Iris data set we observe the presence of 3 clusters in all methods except BMo and perhaps Prop2. For the Wisconsin breast cancer data set we can only observe 2 clear clusters in the Prop2 and BMo results. For the wine recognition data set we observe 3 clear clusters in all results, except in Prop2's result. From these simulations, except for Wisconsin breast cancer data set where Prop2 performs best, it seems that BMe and Prop1 best indicate the correct number of clusters.

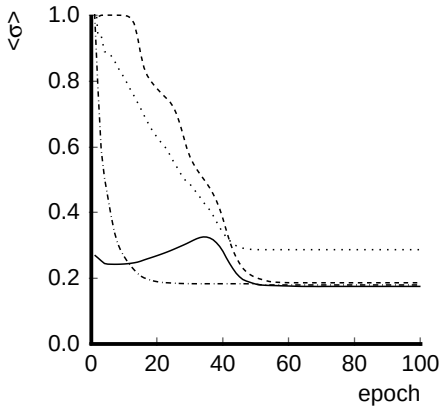

Figure 6: Average kernel radius as a function of training epochs for the update rules shown in Fig. 1, given the Iris data set, and when the neighborhood range vanishes. We observe that Prop1 and Prop2 converge to the solution obtained with GMM. Same line conventions as in Fig. 2.

4 Conclusion

Since there exists no closed form solution for updating the kernel radii in heteroscedastic Gaussian kernel-based topographic map formation, based on an object function, such as log-likelihood maximization or distortion minimization, we suggested several candidate fixed point rules on heuristic grounds. We found that, at least for the data sets considered, the extended batch map algorithm and a new fixed point rule, for which the limiting case is identical to maximum log-likelihood Gaussian mixture modeling, to perform best. We also indicated that some of the rules introduced in the literature are either identical or quite similar,

Proceedings of the 6th International Workshop on Self-Organizing Maps (WSOM 2007)
Published by the Neuroinformatics Group, Bielefeld University, Germany, ISBN 978-3-00-022473-7
All contributions to WSOM 2007 are available online at: http://biecoll.ub.uni-bielefeld.de

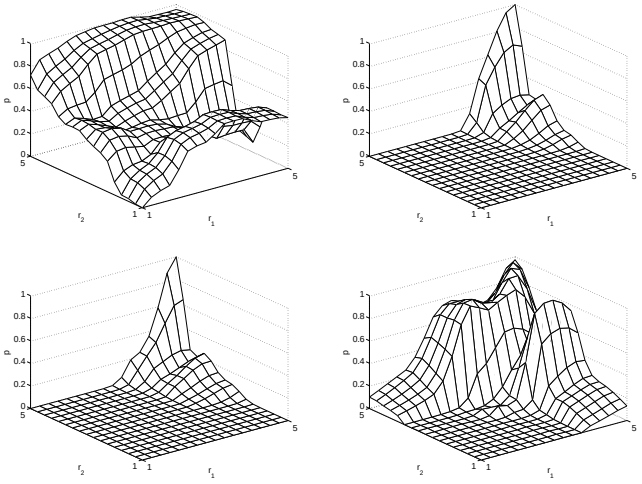

Figure 7: Results for the Wisconsin breast cancer data set. Same conventions as in Fig. 5.

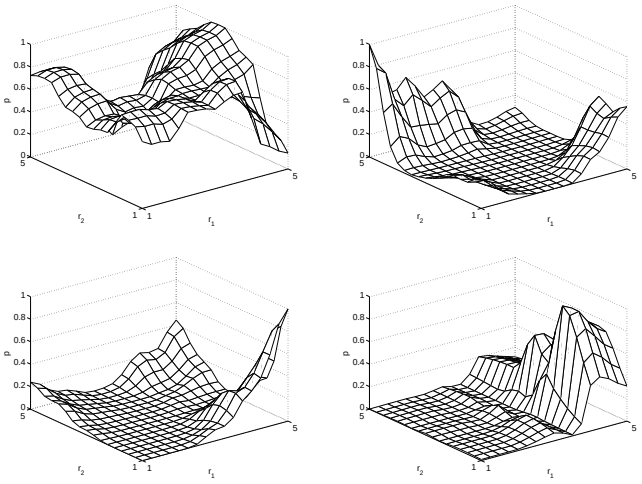

Figure 8: Results for the wine recognition data set. Same conventions as in Fig. 5.

which is an indication that the number of variations on this theme is limited, at least if the rules should also be able to unfold the lattices.

Acknowledgments

The author is supported by the Excellence Financing program of the K.U.Leuven (EF 2005), the Belgian Fund for Scientific Research – Flanders (G.0248.03, G.0234.04), the Flemish Regional Ministry of Education (Belgium) (GOA 2000/11), the Belgian Science Policy (IUAP P5/04), and the European Commission (NEST-2003-012963, STREP-2002-016276, and IST-2004-027017).

References

[1] M. Benaim and L. Tomasini (1991), Competitive and self-organizing algorithms based on the minimization of an information criterion, *Proc. ICANN'91*, pp. 391-396.

[2] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood for incomplete data via the EM algorithm," *J. Roy. Statist. Soc., B*, vol. 39, 1-38, 1977.

[3] T. Graepel, M. Burger, and K. Obermayer, "Self-organizing maps: Generalizations and new optimization techniques," *Neurocomputing*, **vol. 21**, pp. 173-190, 1998.

[4] T. Heskes (2001), Self-organizing maps, vector quantization, and mixture modeling, *IEEE Trans. Neural Networks*, **vol. 12**(6), pp. 1299-1305, 2001.

[5] T. Kohonen, *Self-organizing maps*, Heidelberg: Springer, 1995.

[6] S.P. Luttrell, "Derivation of a class of training algorithms," *IEEE Trans. Neural Networks*, **1**, pp. 229-232, 1990.

[7] R.A. Redner and H.F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review*, vol. 26(2), 195-239, 1984.

[8] A. Ultsch, and H.P. Siemon, "Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis," *Proc. Intl. Neural Networks*, Kluwer Academic Press, Paris, 305-308, 1990.

[9] M.M. Van Hulle, "Joint entropy maximization in kernel-based topographic maps," *Neural Computation*, vol. 14(8), 1887-1906, 2002.

[10] M.M. Van Hulle, "Maximum likelihood topographic map formation," *Neural Computation*, vol. 17(3), pp. 503-513, 2005.

[11] M.M. Van Hulle, "Edgeworth-expanded topographic map formation," in *Proc. WSOM05*, M. Cottrell, Ed., pp. 719-724, 2005

[12] H. Yin and N.M. Allinson (2001), Self-organizing mixture networks for probability density estimation, *IEEE Trans. Neural Networks*, **vol. 12**, pp. 405-411.