KERNEL-BASED TOPOGRAPHIC MAP FORMATION ACHIEVED WITH NORMALIZED GAUSSIAN COMPETITION

Marc M. Van Hulle
K.U.Leuven, Laboratorium voor Neuro- en Psychofysiologie
Campus Gasthuisberg, Herestraat, B-3000 Leuven, BELGIUM
Tel.: + 32 16 34 59 61, Fax: + 32 16 34 59 93
E-mail: marc@neuro.kuleuven.ac.be

Abstract.

A new learning algorithm for kernel-based topographic map formation is introduced. The kernels are Gaussians, and their centers and ranges individually adapted so as to yield an equiprobabilistic topographic map. The converged map also generates a heteroscedastic Gaussian mixture model of the input density. This is verified for both synthetic and real-world examples, and compared with other algorithms for kernel-based topographic map formation.

INTRODUCTION

Kernel-based methods have enjoyed a great deal of attention in the neural network community and are now considered as viable alternatives for supervised learning in tasks such as regression and classification. However, what is less known is that kernels have also been introduced in unsupervised competitive learning, more in particular, in topographic map formation, mostly in an attempt to improve the density estimation properties, the noise tolerance, or even the biological relevance of the popular Self-Organizing Map (SOM) algorithm [1, 2]. The neurons of the topographic map are equipped with local kernel functions (Fig. 1), rather than Winner-Take-All (WTA) functions. We further call such lattices *kernel-based* topographic maps.

Several algorithms for kernel-based topographic map formation are described in the literature (for an overview, see [3]). An early example is the elastic net of Durbin and Willshaw [4], which can be viewed as an equal-variance Gaussian mixture density model, fitted to the data points by a penalized maximum likelihood term. In recent years, algorithms have been introduced by Bishop and co-workers [5] (Generative Topographic Map, based on constrained, equal-variance Gaussian mixture density modeling with equal mixings), Utsugi [6] (also using equal mixings of equal-variance Gaussians), Van Hulle [7, 8] (kernel-based Maximum Entropy learning Rule (kMER),

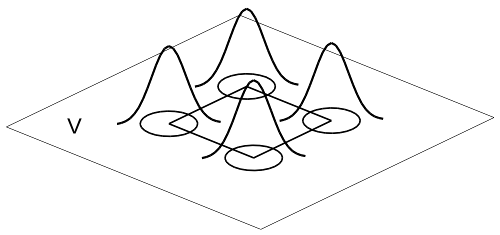

Figure 1: Kernel-based topographic maps. Example of a 2×2 map (*cf.* rectangle in V-space) for which each neuron has a Gaussian kernel as activation- or output function. For each neuron, a circle is drawn in V-space with center the neuron weight vector and radius the kernel range.

which builds equiprobabilistic maps using equal mixtures of Gaussians with differing variances). Furthermore, we should also mention the fuzzy membership in clusters approach of Graepel and co-workers (The Soft Topographic Vector Quantization (STVQ) algorithm) [9] which relies on equal-variance Gaussians. Other approaches are offered by the Local Density Estimation (LDE) algorithm [10], which individually adapts the centers and radii of the Gaussian kernels to the assumed Gaussian local input density, and the joint entropy maximization algorithm [3], which adapts the centers and radii so as to maximize the joint entropy of the kernel outputs. Finally, the original SOM algorithm itself has been regarded as an approximate way to perform equivariance Gaussian mixture density modeling by Yin and Allinson [11], among others (see [3] for references).

In this article, we will introduce a new algorithm for kernel-based topographic map formation, called the Equiprobabilistic Topographic Map (ETM) algorithm, since it is aimed at building equiprobabilistic maps, using Gaussian kernels with differing variances. We will compare the performance of our new learning algorithm with that of several other algorithms, using both synthetic and real-world examples.

EQUIPROBABILISTIC TOPOGRAPHIC MAP FORMATION

Consider a discrete lattice A, with a regular and fixed topology, of arbitrary dimensionality d_A, in d-dimensional input space $V \subseteq \Re^d$. To each of the N nodes of the lattice corresponds a formal neuron, the activation or output of which obeys the Gaussian kernel function:

$$K(\mathbf{v}, \mathbf{w}_i, \sigma_i) = \exp\left(-\frac{\|\mathbf{v} - \mathbf{w}_i\|^2}{2\sigma_i^2}\right), \qquad (1)$$

with $\mathbf{v} \in V$, and with $\mathbf{w}_i$ the kernel center ("weight vector") and σ_i the kernel radius (here, the standard deviation). Depending on the degree of neural activation, the kernel centers are adapted so as to produce a topology-preserving mapping of the input distribution; the kernel radii are adapted so that, on average, each kernel has an equal contribution in modeling the input distribution. The latter is equivalent to having identical activation distributions for these neurons. In order to keep the latter tractable, we adopt for each kernel a common threshold τ on its kernel output and require identical sub- or supra-threshold probabilities (equiprobabilistic map).

Kernel center update

Since we are using kernel functions, rather than Winner-Take-All (WTA) functions, we need a different type of competition between the neurons in order to let their kernels "pave" the input space. Define the fuzzy code membership Ξ_i as the *normalized* Gaussian:

$$\Xi_i(\mathbf{v}) = \frac{K(\mathbf{v}, \mathbf{w}_i, \sigma_i)}{\sum_{k \in A} K(\mathbf{v}, \mathbf{w}_k, \sigma_k)}, \quad \forall i \in A, \tag{2}$$

so that $0 \leq \Xi_i(\mathbf{v}) \leq 1$ and $\sum_i \Xi_i(\mathbf{v}) = 1$. We update the kernel center $\mathbf{w}_i$ proportional to Ξ_i and in the direction of $\mathbf{v}$. Hence, inputs that generate a substantial degree of activation in several neurons – they are "shared" by these neurons – will lead to smaller weight updates. As a result of this, there will be a *competitive* element in the learning process, since the kernel centers will tend to be "pulled" apart by the unbalanced weight update strengths.

Kernel radius update

Besides the kernel centers $\mathbf{w}_i$, we also need to update the kernel radii σ_i. The idea is to adjust them in such a way that the neurons will have identical supra-threshold activation probabilities at convergence. For the threshold, we take the half-height point of the Gaussian kernel's cumulative distribution function:

$$P(\frac{d}{2}, \frac{\|\mathbf{w}_i - \mathbf{v}\|^2}{2\sigma_i^2}) \equiv \frac{\Gamma(\frac{d}{2}, \frac{\|\mathbf{w}_i - \mathbf{v}\|^2}{2\sigma_i^2})}{\Gamma(\frac{d}{2})}, \tag{3}$$

with $\Gamma(.)$ the gamma distribution. Hence, we determine the radius of each kernel for which the previous equation equals $\frac{1}{2}$, on average, given $\mathbf{w}_i$. We will strive towards this solution by incrementally decreasing σ_i when $P(\frac{d}{2}, \frac{\|\mathbf{w}_i - \mathbf{v}\|^2}{2\sigma_i^2})$ $\geq \frac{1}{2}$ for the current input $\mathbf{v}$, and by incrementally increasing it otherwise. For this purpose, we introduce the binary variable $\mathbb{1}_i(\mathbf{v})$:

$$\mathbb{1}_i(\mathbf{v}) = \begin{cases} 1 & \text{if } P(\frac{d}{2}, \frac{\|\mathbf{w}_i - \mathbf{v}\|^2}{2\sigma_i^2}) \geq \frac{1}{2} \\ 0 & \text{if } P(\frac{d}{2}, \frac{\|\mathbf{w}_i - \mathbf{v}\|^2}{2\sigma_i^2}) < \frac{1}{2}, \end{cases} \tag{4}$$

and update σ_i so as to achieve $P(\mathbb{1}_i(z) = 1) = \frac{\rho}{N}$, $\forall i$, with ρ a scale factor (a constant). The larger ρ, the larger the σ_is at convergence will be. In the simulations, we will use $\rho = 1$.

Complete learning algorithm

In order to develop a topology-preserving mapping, we need to supply topological information to the learning process. We will do this in the traditional way by supplementing the kernel center update process with a neighborhood function specified in lattice space coordinates, $\Lambda(i, j, \sigma_\Lambda)$, with σ_Λ the neighborhood range (also in lattice space). Potentially, we could center a neighborhood function around each neuron, since we have a continuously-graded neural activations. However, in order to keep the map formation process computationally tractable, we center the neighborhood function around the most active neuron i^*.[1]

The complete learning algorithm, called Equiprobabilistic Topographic Map (ETM) algorithm, consists of two update rules, one for the kernel centers and another for the kernel radii, and a cooling scheme for the neighborhood function. The on-line[2] update rule for the kernel centers is:

$$\Delta \mathbf{w}_i = \eta \sum_{j \in A} \Lambda(i, i^*, \sigma_\Lambda)\ \Xi_{i^*}(\mathbf{v})(\mathbf{v} - \mathbf{w}_i), \quad \forall i \in A, \tag{5}$$

and that for the kernel radii:

$$\Delta \sigma_i = \eta \left(\frac{\rho_r}{N}(1 - \mathbb{1}_i(\mathbf{v})) - \mathbb{1}_i(\mathbf{v}) \right), \quad \forall i \in A, \tag{6}$$

with $\rho_r \triangleq \frac{\rho N}{N - \rho}$. A batch version of these learning rules can also be derived. As a neighborhood function, we use a Gaussian:

$$\Lambda(i, i^*, \sigma_\Lambda) = \exp\left(- \frac{\|\mathbf{r}_i - \mathbf{r}_{i^*}\|^2}{2\sigma_\Lambda^2} \right), \tag{7}$$

with σ_Λ the neighborhood function range, and $\mathbf{r}_i$ neuron i's lattice coordinate.

Finally, we adopt the following neighborhood cooling scheme:

$$\sigma_\Lambda(t) \quad = \quad \sigma_{\Lambda 0} \exp\left(-2\sigma_{\Lambda 0} \frac{t}{t_{max}} \right), \tag{8}$$

with t the present time step, t_{max} the maximum number of time steps, and $\sigma_{\Lambda 0}$ the range spanned by the neighborhood function at $t = 0$.

LATTICE DISENTANGLING DYNAMICS

The first test one should consider for any topographic map formation algorithm is the ability to disentangle a lattice from a randomly-chosen initial

[1] Note that, usually, the neighborhood function is centered around the neuron for which the weight vector is closest to the current input. This neuron also corresponds to the one that wins the competition in the learning process ("winner").

[2] A batch version of these learning rules can also be derived.

state. We use the standard example of a square lattice and uniform input density. We take a 10×10 lattice and a 2D-uniform input density $[0, 1]^2$. The initial weights are randomly chosen from the same input density. The radii are initialized randomly by sampling the uniform distribution $[0, 0.1]$. We take $t_{max} = 1,000,000$ and $\sigma_{\Lambda 0} = 5$, and keep the learning rate fixed at $\eta_w = 10^{-2}$ and $\eta_\sigma = 0.1\eta_w$. The results are shown in Fig. 2. The radii of the circles correspond to the standard deviations of the corresponding Gaussians.

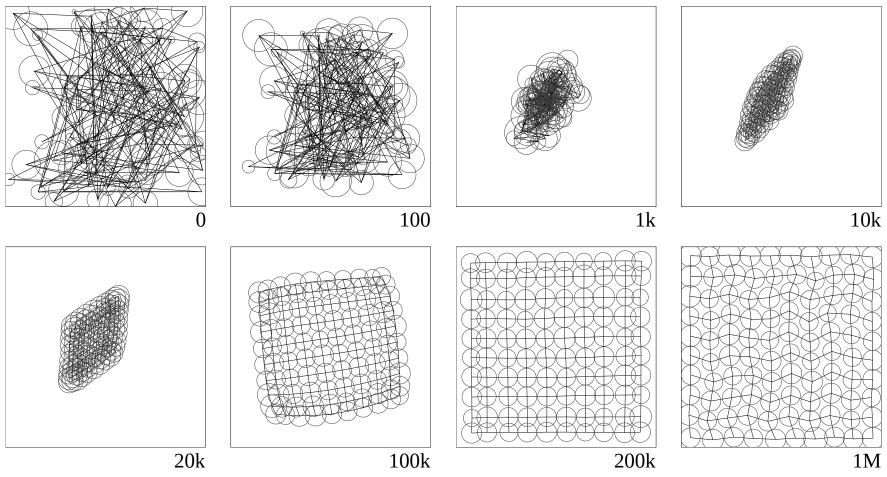

Figure 2: Evolution of a 10×10 lattice. The boxes correspond to the range spanned by the uniform input density. The values given below the boxes represent time.

DENSITY ESTIMATION PERFORMANCE

Since each kernel corresponds to a Gaussian, with mean the corresponding neuron's kernel center and standard deviation the kernel radius, and since the kernels are equiprobabilistic, we can estimate the input density $p(\mathbf{v})$ in a way similar to the classic *Variable Kernel* density estimation method [13]:

$$\widehat{p}(\mathbf{v}) = \sum_{i=1}^{N} \frac{\exp(-\frac{\|\mathbf{v}-\mathbf{w}_i\|^2}{2(\sigma_i)^2})}{N(2\pi)^{\frac{d}{2}}\sigma_i^d}, \tag{9}$$

namely, in terms of an equal mixture of Gaussians with differing variances. In order to assess the density estimation performance of our algorithm, we will consider two synthetic and one real-world example. Furthermore, we will also run several other kernel-based topographic map formation algorithms and compare their density estimation performances: the algorithm of Yin and Allinson [11], the STVQ and SSOM algorithms [9], the kMER algorithm [7], the LDE algorithm [10], and the joint entropy maximization (max(JE)) algorithm [3].

Synthetic examples

We train a 9 neuron, 1D-lattice in 1D-space and take $\eta_w = 0.01$, $\eta_\sigma = 0.1\eta_w$, $t_{max} = 100,000$ and $\sigma_{\Lambda 0} = 4.5$. Note that, albeit the input is Gaussian, the distribution of inputs that activate a given kernel will be non-Gaussian, since the kernels will occupy different positions along the real line.

The density estimate $\widehat{p}(\mathbf{v})$ and the original Gaussian distribution $p(\mathbf{v})$ are shown in Fig. 3A. The mean squared error (MSE) between $\widehat{p}(\mathbf{v})$ and $p(\mathbf{v})$ is $2.34\ 10^{-4}$ when calculated for 100 uniform positions in the range shown. (Note that, instead of the MSE, we could also use the Kullback-Leibler divergence here.) In the case of the other algorithms mentioned, we run all simulations on the same input data, using the cooling scheme, except for the SSOM and STVQ algorithms. For the STVQ algorithm, we take for the neighborhood function radius $\sigma_\Lambda = 0.5$, and for the equal and constant kernel radii $\frac{1}{\beta} = 0.01$, as suggested in [9]. We also adopt these parameter values for the SSOM algorithm, since it is in fact a limiting case of the STVQ algorithm. Furthermore, again for the SSOM and STVQ algorithms, since they do not adapt their kernel radii, we look for the (common) kernel radius that optimizes the MSE between the estimated and the theoretical distributions. In this way, we at least know that a better MSE result cannot be obtained. We also optimize the ρ_s parameter of kMER in this way, for the same reason. For comparison's sake, we also show the density estimate for the kMER algorithm, since it is also aimed at building equiprobabilistic topographic maps (Fig. 3B). The MSE results of all algorithms considered are summarized in Table 1, together with the algorithms' parameters. We observe that most algorithms yield a similar performance.

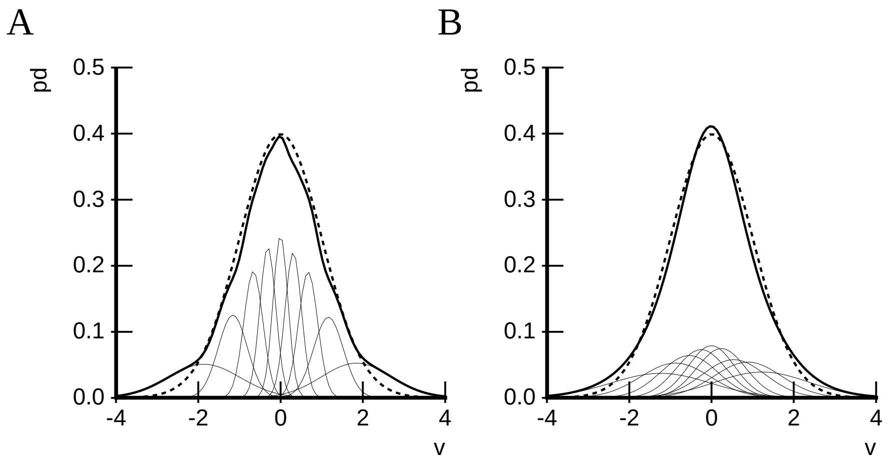

Figure 3: One-dimensional, unit-variance Gaussian input density (thick dashed line) and the estimates (thick continuous lines) obtained with our algorithm ETM (A) and kMER (B), when 9 kernels are used (thin continuous lines). Abbreviation: pd = probability density.

Table 1: Density estimation performance (in MSE) for the standard normal distribution for different algorithms. The algorithms' parameter settings are also given.

Algorithm	Parameters	MSE
ETM	$\eta_w = 0.01$, $\eta_\sigma = 0.1\eta_w$	$2.34\ 10^{-4}$
Yin & Allinson	$\eta_w = 0.001$, $\eta_\sigma = 0.1\eta_w$	$1.18\ 10^{-4}$
STVQ	$\frac{1}{\beta} = 0.01$, $\sigma_\Lambda = 0.5$, $\sigma_{\text{opt}} = 0.555$	$1.99\ 10^{-3}$
SSOM	$\frac{1}{\beta} = 0.01$, $\sigma_\Lambda = 0.5$, $\sigma_{\text{opt}} = 0.520$	$1.99\ 10^{-3}$
kMER	$\eta_w = 0.001$, $\rho_r = 1$, $\rho_s = 4.58$	$1.75\ 10^{-4}$
LDE	$\eta_w = 0.01$, $\eta_\sigma = 0.1\eta_w$	$7.05\ 10^{-4}$
max(JE)	$\eta_w = 0.001$, $\eta_\sigma = 0.1\eta_w$	$9.35\ 10^{-5}$

For the second example, we take a skewed distribution consisting of a mixture of two Gaussians, G$(0, 1)$ and G$(1.5, 1/3)$, with mixing coefficients 3/4 and 1/4. We re-run the algorithms using the same settings as in the first example. The resulting MSE equals $6.99\ 10^{-5}$, for the ETM algorithm, and $5.49\ 10^{-4}$ for kMER. The corresponding density estimates are shown in Fig. 4A,B, respectively. The MSE results for all algorithms are listed in Table 2, from which we can verify that the ETM algorithm performs best.

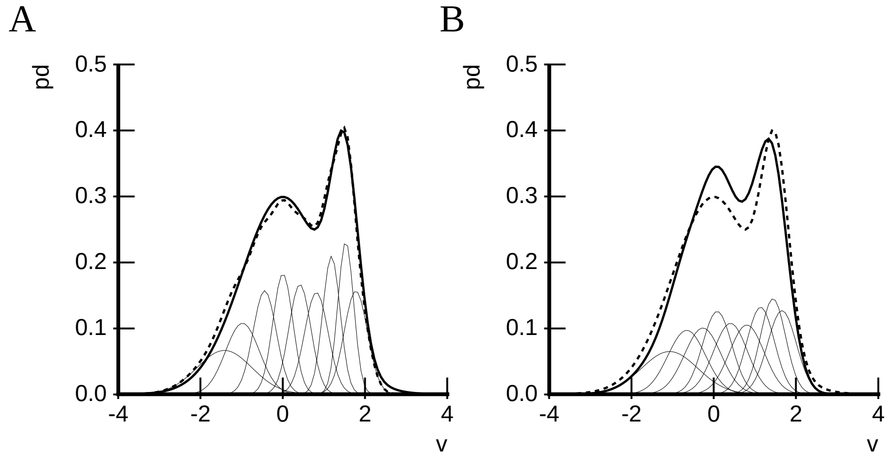

Figure 4: One-dimensional mixture of two Gaussians, with differing radii and mixing coefficients. Same conventions as in Fig. 3.

Real-world example

The example is a classic benchmark in the density estimation literature. The data set consists of 222 observations of eruption lengths of the Old Faithful geyser in Yellowstone National Park, and was compiled by Weisberg [13]. Since we now do not dispose of the theoretical distribution, we only show the density estimates (Fig. 5), except for the SSOM algorithm since the result is quite similar to that of the STVQ algorithm.

Table 2: Density estimation performance for a skewed bimodal distribution consisting of two Gaussians.

Algorithm	Parameters	MSE
ETM	$\eta_w = 0.01$, $\eta_\sigma = 0.1\eta_w$	$6.99\ 10^{-5}$
Yin & Allinson	$\eta_w = 0.001$, $\eta_\sigma = 0.1\eta_w$	$9.06\ 10^{-4}$
STVQ	$\frac{1}{\beta} = 0.01$, $\sigma_\Lambda = 0.5$, $\sigma_{\text{opt}} = 0.495$	$2.17\ 10^{-3}$
SSOM	$\frac{1}{\beta} = 0.01$, $\sigma_\Lambda = 0.5$, $\sigma_{\text{opt}} = 0.450$	$1.88\ 10^{-3}$
LDE	$\eta_w = 0.01$, $\eta_\sigma = 0.1\eta_w$	$7.17\ 10^{-4}$
kMER	$\eta_w = 0.001$, $\rho_r = 1$, $\rho_s = 2.32$	$5.49\ 10^{-4}$
max(JE)	$\eta_w = 0.001$, $\eta_\sigma = 0.1\eta_w$	$4.18\ 10^{-4}$

Furthermore, for comparison's sake, we also consider a more traditional variable kernel density estimation method, namely, the adaptive unbiased cross-validation (adaptive UCV) method [14], which allocates a kernel at each data point. The adaptive UCV method has been shown to yield particularly good results, compared to other traditional methods, on the Old Faithful geyser data set [14]. The adaptive UCV result is superimposed on the density estimates shown in Fig. 5. We observe that the ETM algorithm best approximates the UCV result for the valley between two peaks, as well as their locations. However, one should be aware of the fact that the UCV result is in itself an approximation of the unknown true density distribution.

CONCLUSION

We have introduced a new learning algorithm for kernel-based topographic map formation, called Equiprobabilistic Topographic Map (ETM) algorithm. The kernels are Gaussians, and their centers and ranges are individually adapted so as to yield an equiprobabilistic topographic map (whence the algorithm's name). The converged map also yields a density estimate in terms of an equal mixture of heteroscedastic Gaussians. We have compared the density estimation performance of our algorithm to that of several other kernel-based topographic map formation algorithms. The results show that our algorithm performs best, or is among the best performing algorithms.

Acknowledgments

M.M.V.H. is supported by research grants received from the Fund for Scientific Research (G.0185.96N), the National Lottery (Belgium) (9.0185.96), the Flemish Regional Ministry of Education (Belgium) (GOA 95/99-06; 2000/11), the Flemish Ministry for Science and Technology (VIS/98/012), and the European Commission (QLG3-CT-2000-30161 and IST-2001-32114). The author is solely responsible for the contents of this article. It does not represent the opinion of the Community, which is also not responsible for any use that might be made of data appearing therein.

REFERENCES

[1] T. Kohonen, "Self-organized formation of topologically correct feature maps," Biol. Cybern., vol. 43, pp. 59-69, 1982.

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

[3] M.M. Van Hulle, "Joint entropy maximization in kernel-based topographic maps," Neural Computat., vol. 14, 2002, in press.

[4] R. Durbin and D. Willshaw "An analogue approach to the travelling salesman problem using an elastic net method" Nature, vol. 326, pp. 689-691, 1987.

[5] C.M. Bishop, M. Svensén, and C.K.I. Williams, "GTM: The generative topographic mapping. Neural Computat., vol. 10, pp. 215-234, 1998.

[6] A. Utsugi "Hyperparameter selection for self-organizing maps," Neural Computat., vol. 9, pp. 623-635, 1997.

[7] M.M. Van Hulle, "Kernel-based equiprobabilistic topographic map formation," Neural Computation, vol. 10, pp. 1847-1871, 1998.

[8] M.M. Van Hulle Faithful representations and topographic maps: From distortion- to information-based self-organization, New York: Wiley, 2000.

[9] T. Graepel, M. Burger, and K. Obermayer, "Phase transitions in stochastic self-organizing maps," Physical Review E, vol. 56(4), pp. 3876-3890, 1997.

[10] M.M. Van Hulle, "Kernel-based topographic map formation by local density modeling," Neural Computat., vol. 14, 2002, in press.

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

[12] T. Kostiainen and J. Lampinen, "Generative probability density model in the self-organizing map," Self-organizing neural networks: Recent advances and applications, U. Seiffert & L. Jain (Eds.), pp. 75-94. Heidelberg: Physica Verlag, 2002.

[13] B.W. Silverman, Density Estimation for Statistics and Data Analysis, Chapman and Hall: London, 1992.

[14] S.R. Sain and D.W. Scott, "On Locally Adaptive Density Estimation" J. American Statistical Association, vol. 91, pp. 1525-1534, 1996.

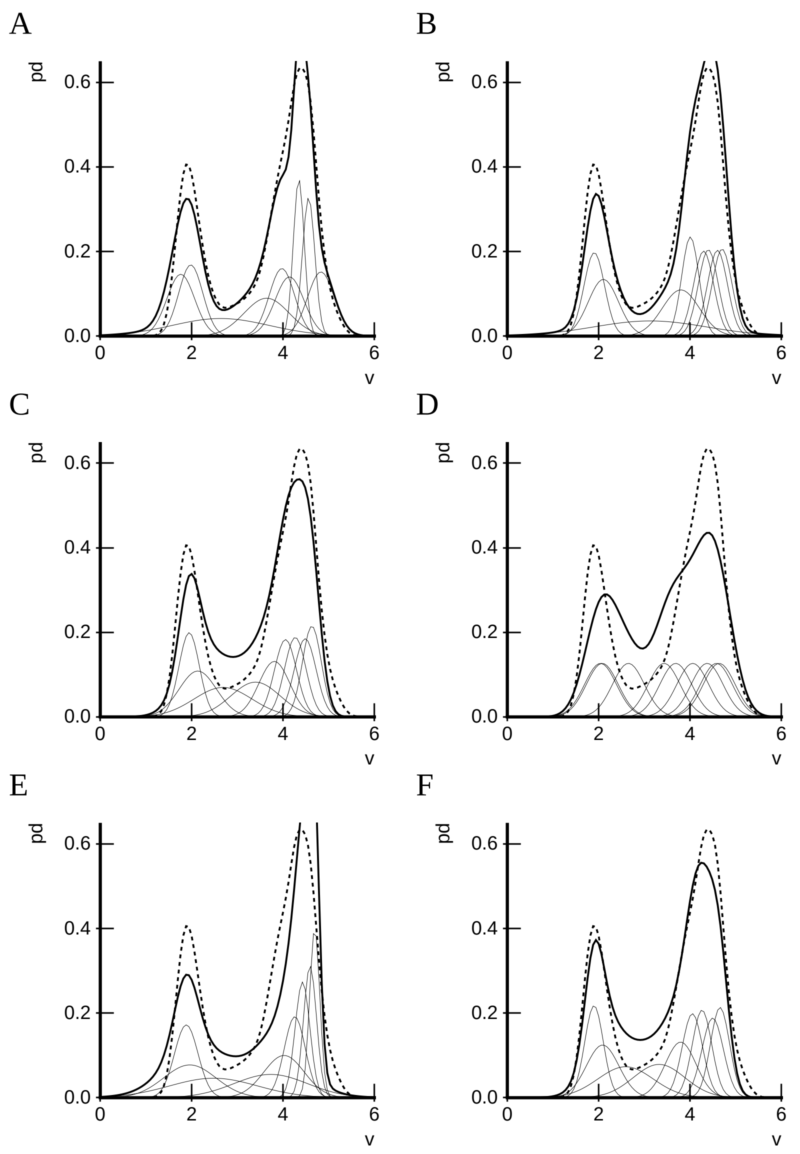

Figure 5: Density estimates obtained for the Old Faithful geyser eruption lengths data set with our ETM algorithm (A), the kMER algorithm (B), Yin and Allinson's algorithm (C), and the STVQ (D), LDE (E), and max(JE) algorithms (F) (thick continuous lines). The $N = 9$ converged kernels (thin continuous lines), as well as the estimate obtained with the more traditional adaptive UCV method (thick dashed line), are also shown in each of these plots.