International Journal of Neural Systems, Vol. 24, No. 4 (2014) 1450016 (18 pages)

DOI: 10.1142/S0129065714500166

World Scientific
www.worldscientific.com

BREGMAN DIVERGENCES FOR GROWING HIERARCHICAL SELF-ORGANIZING NETWORKS

EZEQUIEL LÓPEZ-RUBIO*, ESTEBAN JOSÉ PALOMO†
and ENRIQUE DOMÍNGUEZ‡
Department of Computer Science
University of Malaga, 29071, Spain
**ezeqlr@lcc.uma.es*
†*ejpalomo@lcc.uma.es*
‡*enriqued@lcc.uma.es*

Accepted 24 February 2014
Published Online 25 March 2014

Growing hierarchical self-organizing models are characterized by the flexibility of their structure, which can easily accomodate for complex input datasets. However, most proposals use the Euclidean distance as the only error measure. Here we propose a way to introduce Bregman divergences in these models, which is based on stochastic approximation principles, so that more general distortion measures can be employed. A procedure is derived to compare the performance of networks using different divergences. Moreover, a probabilistic interpretation of the model is provided, which enables its use as a Bayesian classifier. Experimental results are presented for classification and data visualization applications, which show the advantages of these divergences with respect to the classical Euclidean distance.

Keywords: Bregman divergences; self organization; classification; visualization.

1. Introduction

The self-organizing map (SOM) is a very popular neural network model for data analysis and visualization of high-dimensional input data. SOM is trained using unsupervised learning to produce a two-dimensional discretized representation of input space of the training cases. The SOM[1] has shown to be exceptionally successful in mapping high-dimensional input data to a two-dimensional output space such that similar inputs are mapped onto neighboring regions of the map. In other words, the similarity of the input data is preserved as faithfully as possible within the representation space of the SOM. However, its static nature in terms of map size and the fact that the size must be determined prior to training usually implies multiple runs until optimal results are reached. Especially when large data sets are to be clustered at a fine-grained map resolution, i.e. the map consists of a large number of units, this can be a very time consuming task. Furthermore, the two-dimensional map representation inherently disregards potential hierarchical characteristics of the data.

Several related unsupervised neural models have been proposed to enhance the practicability of the SOM.[2–6] Different modifications of the SOM suggest different enhancements from different viewpoints. SOM-based neural networks that grow during map training have in common that the training starts with a rather small number of units (neurons). New neurons are inserted into the network at certain iterations until a stopping criterion. In some models,

*Corresponding author.

links between neurons are being added or removed during training, thus, influencing their neighborhood relations and allowing for a stronger separation of clusters. For instance, the incremental grid growing (IGG)[7] initially consists of four connected neurons in a rectangular grid structure. During the training process, the structure as well as the connectivity of the network is dynamically adapted by adding new neurons at the border of the network adjacent to the neuron having the maximum quantization error, in order to provide more map space for a better representation of the input data. Similarly, during the training of the growing cell structures (GCS),[8] neurons are added and the state of connections changed, but with more freedom regarding the topology of the map space. Growing neural gas (GNG)[9,10] uses a similar algorithm but implements a different learning rule. In the growing grid (GG),[11] complete rows and columns of neurons are added to the network maintaining a rectangular grid until the training process is terminated. New neurons are inserted between the neuron with the highest number of hits and its most dissimilar neighbor in terms of weight vector distance, while the connections between the neurons remain untouched. A growing grid variant with an adaptive hyper-cubical output space is presented in Ref. 12.

Other type of SOM variants are hierarchical models such as the hierarchical feature map (HFM).[13] This model consists of a pyramidal hierarchy of SOMs with a map at the top layer. In the second layer, a map is presented for each neuron at the top layer and this principle is repeated with any further layers. Map training is performed top-down according to the standard SOM training algorithm, but the size of the single maps and the hierarchical structure itself has to be determined prior to training. The evolving tree[14] is other SOM-based network overcoming the constraints of map-like topologies with the neurons being arranged in a growing tree topology. Among the large number of research publications discussing the SOM and its different variants and extensions, one of the SOM-based models implementing an algorithm dealing with both disadvantages of fixed size and missing hierarchical representation is the growing hierarchical self-organizing map (GHSOM).[15] The GHSOM combines the advantages of the two principal extensions of the SOM, dynamic growth and hierarchical structure. Basically, this neural model is composed of independent growing SOMs, which are arranged in layers. The maps are allowed to grow in size during the training process until a certain quality criterion, and this growth process is further continued to form a layered architecture such that hierarchical relations between the input data are further detailed at lower layers of the hierarchy. Consequently, the number of layers, maps and neurons are determined during the training process, that is, the architecture is automatically adapted itself according to the structure of the input space during the training process until a certain quality criterion.

An obvious criterion to guide the training process is the quantization error $\mathcal{E}_i$, calculated as the sum of the distances between the weight vector of a unit i and the input vectors mapped onto this unit. It is used to evaluate the mapping quality of a SOM that is based on the mean quantization error ($\mathcal{E}$) of all units on the map. A map grows until its $\mathcal{E}$ is reduced to a certain fraction τ_1 of $\mathcal{E}_i$ of unit i in the preceding layer of the hierarchy. The neural architecture is expanded by another layer in case of dissimilar input data being mapped on a particular unit. These units are identified by a rather high quantization error $\mathcal{E}_i$ which is above a threshold τ_2. This threshold basically indicates the desired granularity of data representation as a fraction of the initial quantization error at the top layer. Briefly, the growth process of the GHSOM works according to the values of τ_1 and τ_2. The smaller the parameter value τ_1, the larger maps and the more shallow the hierarchy, and that the lower the setting of parameter τ_2, the larger the number of layers. In this sense, the threshold of these parameters gives a criterion to hierarchies. Another neural network based on GHSOM with parameter self-adjustment focusing on robustness regarding clustering of non-stationary data is the dynamic adaptive self-organizing hybrid (DASH) model.[16] Other alternative strategies and quality measures were analyzed in Ref. 17. In this work, authors propose a new strategy based on the determination of desired levels of the hierarchy, but an additional parameter is introduced.

In this work, Bregman divergences are introduced to develop a new self-organizing network model, that we call growing hierarchical Bregman self-organizing map (GHBSOM). The main reason for this choice is that Bregman divergences are the only ones whose

minimizer is the mean, and this makes them particularly suitable to growing hierarchical networks. This way we open a new application field for Bregman divergences, since they have been used in the past for flat topology SOMs only.

The remainder of this paper is organized as follows. Section 2 presents and explains the proposed model. Section 3 provides a discussion about the most outstanding features of the GHBSOM. Experimental results with all Bregman divergences are shown in Sec. 4. Finally, some conclusions are drawn in Sec. 5.

2. Model

A GHBSOM network is defined as a hierarchy of SOMs. Each SOM is a rectangular lattice of units; its size (number of rows and columns) can grow as the map learns. Every map is the child of a unit in the upper layer, except for the top level (root) map. The root map contains only one unit with the overall mean of the input dataset; it does not learn nor does it grow. The general structure of a GHBSOM network is depicted in Fig. 1.

First the fundamentals of Bregman divergences are reviewed (Sec. 2.1). After that the normal operation of a SOM when it does not grow is explained (Sec. 2.2). Then we define how the maps grow (Sec. 2.3) and when a unit produces a child in the lower layer (Sec. 2.4). Section 2.5 presents the proposed learning algorithm. Section 2.6 introduces a probabilistic interpretation of the model, and finally Sec. 2.7 addresses the problem of how to compare the performance of networks with different Bregman divergences.

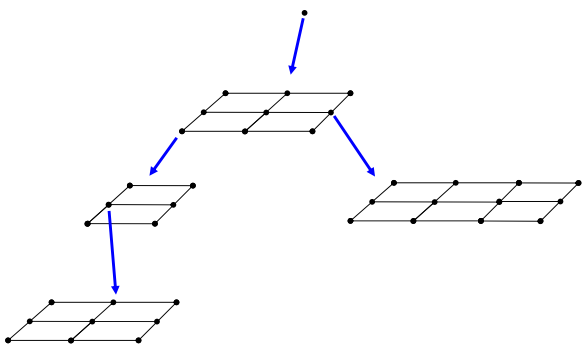


Fig. 1. Structure of a GHBSOM network.

2.1. *Review of Bregman divergences*

Next the fundamentals of Bregman divergences and their application to clustering are reviewed. Let $\phi : \mathcal{S} \to \mathbb{R}$ be a strictly convex real valued function defined over a convex set $\mathcal{S} \subseteq \mathbb{R}^D$, where D is the dimension of the input data.[18–20] We assume that ϕ is differentiable on the relative interior $\mathrm{ri}(\mathcal{S})$ of the set $\mathcal{S}$.[21] Then the Bregman divergence $D_\phi : \mathcal{S} \times \mathrm{ri}(\mathcal{S}) \to [0, +\infty)$ corresponding to ϕ is defined as:

$$D_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - (\mathbf{x} - \mathbf{y})^T \nabla\phi(\mathbf{y}), \quad (1)$$

where $\mathbf{x} \in \mathcal{S}$ and $\nabla\phi(\mathbf{y})$ stands for the gradient vector of ϕ evaluated at $\mathbf{y} \in \mathrm{ri}(\mathcal{S})$. Table 1 lists the Bregman divergences that we consider in this paper.

Bregman divergences are suited for clustering because their minimizer is the mean. This is the main contribution of Ref. 21, where it is proved that the class of distortion measures with respect to a set of centroids which admit an iterative minimization procedure is precisely that of Bregman divergences. Moreover, it is also proved that each Bregman divergence is uniquely associated to a regular exponential family of probability density functions, that are defined below. This way, a unique probability density function can be linked to the cluster associated to a given centroid, which enables probabilistic soft clustering. Moreover, expectation maximization can be carried out with a reduced computational complexity for general Bregman divergences, so that specific Bregman divergences can be designed to suit the application at hand.

The property that the mean is the minimizer of a Bregman divergence is formalized next. Given an input distribution for $\mathbf{x}$ the following condition holds[20]:

$$\boldsymbol{\mu} = E[\mathbf{x}] = \arg\min_{\mathbf{y}} E[D_\phi(\mathbf{x}, \mathbf{y})]. \quad (2)$$

Let N be the number of clusters, and let $\boldsymbol{\mu}_i$ be the mean vector of the ith cluster $\mathcal{C}_i$, $i \in \{1, \ldots, N\}$. Then a point $\mathbf{x}$ belongs to $\mathcal{C}_i$ if $\boldsymbol{\mu}_i$ minimizes the divergence with respect to $\mathbf{x}$:

$$\mathcal{C}_i = \left\{ \mathbf{x} \in \mathcal{S} \,|\, i = \arg\min_{j \in \{1,\ldots,N\}} D_\phi(\mathbf{x}, \boldsymbol{\mu}_j) \right\}. \quad (3)$$

So, we can rewrite (2) to partition $\mathcal{S}$ into N clusters $\mathcal{C}_i$:

$$\boldsymbol{\mu}_i = E[\mathbf{x} \,|\, \mathcal{C}_i] = \arg\min_{\mathbf{y}} E[D_\phi(\mathbf{x}, \mathbf{y}) \,|\, \mathcal{C}_i]. \quad (4)$$

Table 1. Bregman divergences considered in this paper. $\mathbb{R}_+^D$ stands for the set of vectors of size D with strictly positive real components.

Divergence	$\mathcal{S}$	$\phi(\mathbf{x})$	$D_\phi(\mathbf{x},\mathbf{y})$
Squared Euclidean distance	$\mathbb{R}^D$	$\|\mathbf{x}\|^2$	$\|\mathbf{x}-\mathbf{y}\|^2$
Generalized I-divergence	$\mathbb{R}_+^D$	$\sum_{k=1}^D x_k \log x_k$	$\sum_{k=1}^D (-x_k + y_k + x_k \log \frac{x_k}{y_k})$
Itakura–Saito distance	$\mathbb{R}_+^D$	$-\sum_{k=1}^D \log x_k$	$\sum_{k=1}^D (-1 + \frac{x_k}{y_k} - \log \frac{x_k}{y_k})$
Exponential loss	$\mathbb{R}^D$	$\sum_{k=1}^D \exp x_k$	$\sum_{k=1}^D (\exp x_k - \exp y_k - (x_k - y_k)\exp y_k)$
Logistic loss	$(0,1)^D$	$\sum_{k=1}^D (x_k \log x_k + (1-x_k)\log(1-x_k))$	$\sum_{k=1}^D (x_k \log \frac{x_k}{y_k} + (1-x_k)\log \frac{1-x_k}{1-y_k})$

A special kind of Bregman divergences are the regular ones, which are defined as those such that the convex conjugate ψ of ϕ is strictly convex:

$$\psi(\mathbf{z}) = \sup_{\mathbf{x}\in\mathcal{S}} \{\mathbf{z}^T\mathbf{x} - \phi(\mathbf{x})\}. \tag{5}$$

The importance of regular Bregman divergences lies in the fact that they are associated to a probability density function of the following form:

$$p_\phi(\mathbf{x}) = \exp(-D_\phi(\mathbf{x},\boldsymbol{\mu}))\beta_\phi(\mathbf{x}), \tag{6}$$

where β_ϕ is a uniquely determined function which depends on ϕ only, and not on the mean vector $\boldsymbol{\mu}$.[22]

2.2. *Basic model*

For clustering and self-organizing network applications it is necessary to learn a weight vector $\mathbf{w}_i$ of each cluster i,[23] so that $\mathbf{w}_i$ estimates the cluster mean vector $\boldsymbol{\mu}_i$. Stochastic gradient descent has been proposed in Ref. 20 to minimize $E[D_\phi(\mathbf{x},\mathbf{z})]$:

$$\triangle\mathbf{w}_i = -\epsilon\frac{\partial D_\phi(\mathbf{x},\mathbf{w}_i)}{\partial\mathbf{w}_i}, \tag{7}$$

where ϵ is a suitable *step size.*

Here we propose a different approach, namely the estimation of the cluster mean vector $E[\mathbf{x}\,|\,\mathcal{C}_i]$ by stochastic approximation.[24–27] This strategy has been successfully applied by the authors to other self-organizing models in Refs. 28–30. The goal of stochastic approximation is to find the value of some parameter $\boldsymbol{\theta}$ which satisfies

$$\zeta(\boldsymbol{\theta}) = 0, \tag{8}$$

where ζ is a function whose values cannot be obtained directly. What we have is a random variable z which is a noisy estimate of ζ:

$$E[z(\boldsymbol{\theta})\,|\,\boldsymbol{\theta}] = \zeta(\boldsymbol{\theta}). \tag{9}$$

Under these conditions, the Robbins–Monro algorithm[31] proceeds iteratively:

$$\boldsymbol{\theta}(n+1) = \boldsymbol{\theta}(n) + \epsilon(n)z(\boldsymbol{\theta}(n)), \tag{10}$$

where n is the time step.

In our case, the varying parameter $\boldsymbol{\theta}(n)$ is the ith weight vector:

$$\boldsymbol{\theta}(n) = \mathbf{w}_i(n). \tag{11}$$

As said before, we aim to estimate the conditional expectation $\boldsymbol{\mu}_i = E[\mathbf{x}\,|\,\mathcal{C}_i]$ by stochastic approximation, so we may take

$$\zeta(\mathbf{w}_i) = \boldsymbol{\mu}_i - \mathbf{w}_i, \tag{12}$$

$$z(\mathbf{w}_i) = \frac{\mathbb{I}(\mathbf{x}\in\mathcal{C}_i)}{P(\mathcal{C}_i)}(\mathbf{x}-\mathbf{w}_i), \tag{13}$$

where $\mathbb{I}$ stands for the indicator function and $P(\mathcal{C}_i)$ is the *a priori* probability of cluster $\mathcal{C}_i$. Please note that

$$\mathbf{x}\notin\mathcal{C}_i \Rightarrow z(\mathbf{w}_i) = \mathbf{0}. \tag{14}$$

Consequently, we have that (13) satisfies the condition (9):

$$\begin{aligned} E[z(\mathbf{w}_i)\,|\,\mathbf{w}_i] &= P(\mathcal{C}_i)E[z(\mathbf{w}_i)\,|\,\mathcal{C}_i,\mathbf{w}_i] \\ &\quad + P(\bar{\mathcal{C}}_i)E[z(\mathbf{w}_i)\,|\,\bar{\mathcal{C}}_i,\mathbf{w}_i] \\ &= E[\mathbf{x}-\mathbf{w}_i\,|\,\mathcal{C}_i,\mathbf{w}_i] \\ &= E[\mathbf{x}\,|\,\mathcal{C}_i] - \mathbf{w}_i \\ &= \boldsymbol{\mu}_i - \mathbf{w}_i, \end{aligned} \tag{15}$$

where $\bar{\mathcal{C}}_i$ is the complement of cluster $\mathcal{C}_i$.

Hence Eq. (10) reads

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) + \epsilon(n) z(\mathbf{w}_i(n)). \tag{16}$$

If we take

$$\epsilon(n) = P(\mathcal{C}_i)\eta(n) \tag{17}$$

then (16) can be rewritten as:

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) + \eta(n) P(\mathbf{x}(n) \in \mathcal{C}_i) \times (\mathbf{x}(n) - \mathbf{w}_i(n)), \tag{18}$$

where $\eta(n)$ is the learning rate at time step n, and we no longer need the value of the *a priori* probability $P(\mathcal{C}_i)$. Since the exact value of the cluster mean $\boldsymbol{\mu}_i$ is not known, we cannot evaluate $\mathbb{I}(\mathbf{x}(n) \in \mathcal{C}_i)$. Hence we estimate the term $\mathbb{I}(\mathbf{x}(n) \in \mathcal{C}_i)$ by Heskes' confusion probability,[32] which introduces self-organization into the model:

$$P(\mathbf{x}(n) \in \mathcal{C}_i) = \Lambda(i, \text{Winner}(n)), \tag{19}$$

$$\text{Winner}(n) = \arg \min_{j \in \{1,\ldots,N\}} D_\phi(\mathbf{x}(n), \mathbf{w}_j(n)), \tag{20}$$

where Λ is the neighborhood function, and $\text{Winner}(n)$ is the index of the winning neuron. The neighborhood function Λ varies with the time-step n and depends on the topological distance d and a decaying *neighborhood radius* $\Delta(n)$:

$$\Lambda(i, \text{Winner}(\mathbf{x}(n))) = \exp\left(-\left(\frac{d(i, \text{Winner}(\mathbf{x}(n)))}{\Delta(n)}\right)^2\right), \tag{21}$$

$$\Delta(n+1) \leq \Delta(n). \tag{22}$$

Equation (20) leads to the update equation for the SOM:

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) + \eta(n)\Lambda(i, \text{Winner}(n)) \times (\mathbf{x}(n) - \mathbf{w}_i(n)), \tag{23}$$

which equals that of the original Kohonen's SOM if and only if the divergence D_ϕ is the Euclidean distance.

2.3. *Map growth*

The intuition behind map growth is that the performance of a map which does not accurately represent the input data could be improved by enlarging it. To formalize this intuition, the first step is the development of an error measure. The quantization error for an input sample $\mathbf{x}$ is defined as:

$$\mathcal{E}(\mathbf{x}) = \min_{i \in \{1,\ldots,N\}} D_\phi(\mathbf{x}, \mathbf{w}_i). \tag{24}$$

The quantization error for a unit i is the average quantization error over its cluster $\mathcal{C}_i$:

$$\mathcal{E}_i = E[\mathcal{E}(\mathbf{x}) \,|\, \mathcal{C}_i]. \tag{25}$$

Finally, the map quantization error is obtained by averaging over all its units:

$$\mathcal{E} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{E}_i. \tag{26}$$

We decide whether a map must grow as follows. Let $\mathcal{E}_i$ be the quantization error of the parent unit of a map with quantization error $\mathcal{E}$. If the following condition holds, then the map is selected for growing:

$$\mathcal{E} > \tau_1 \mathcal{E}_i, \tag{27}$$

where $\tau_1 \in (0, 1)$ is a tunable parameter. This means that the map grows if its quantization error is too high when compared to that of its parent unit. The parameter τ_1 controls the growth: if τ_1 is low then the map grows more often, and *vice versa*.

The growth is carried out by adding either a row or a column to the map. First we compute the unit j with the highest error:

$$j = \arg \max_{i \in \{1,\ldots,N\}} \mathcal{E}_i. \tag{28}$$

Then we find the immediate neighbor of j with the highest error:

$$k = \arg \max_{i \in \text{Neighbors}(j)} \mathcal{E}_i. \tag{29}$$

After that, we add a line of neurons between j and k, as shown in Fig. 2. This is aimed to reduce

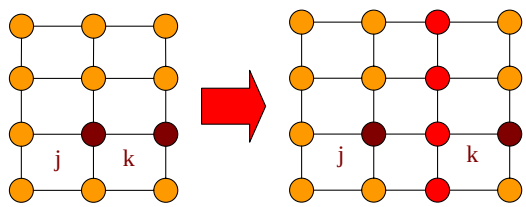


Fig. 2. Map growth. A column of units is inserted between the unit j with the highest error and its immediate neighbor with the highest error k.

the error in the region of the map where the units with the highest error lie.

Each new unit between two old units a and b should represent the union of the clusters of the two old units:

$$\begin{aligned}\boldsymbol{\mu}_{\text{new}} &= E[\mathbf{x} \mid \mathcal{C}_a \cup \mathcal{C}_b] \\ &= P(\mathcal{C}_a)E[\mathbf{x} \mid \mathcal{C}_a] + P(\mathcal{C}_b)E[\mathbf{x} \mid \mathcal{C}_b] \\ &= P(\mathcal{C}_a)\boldsymbol{\mu}_a + P(\mathcal{C}_b)\boldsymbol{\mu}_b. \end{aligned} \tag{30}$$

Hence the initialization of the new weight vector reads

$$\mathbf{w}_{\text{new}} = P(\mathcal{C}_a)\mathbf{w}_a + P(\mathcal{C}_b)\mathbf{w}_b. \tag{31}$$

The estimation of the *a priori* probabilities in (31) can be done by keeping a running sum of the values of the neighborhood function for each unit. However, we have found that in practice we may assume that $P(\mathcal{C}_a) \approx P(\mathcal{C}_b)$ to obtain a faster initialization:

$$\mathbf{w}_{\text{new}} = \frac{1}{2}(\mathbf{w}_a + \mathbf{w}_b). \tag{32}$$

2.4. *Map creation*

Now we must give a criterion to expand a unit of a map into a child map. Here we are interested in reaching a situation where all the leaf units, i.e. those without child maps, have a small quantization error when compared to the initial error $\mathcal{E}_{\text{root}}$ of the root unit. In order to achieve this, we expand a unit i when its quantization error $\mathcal{E}_i$ fulfils the following condition:

$$\mathcal{E}_i > \tau_2 \mathcal{E}_{\text{root}}, \tag{33}$$

where $\tau_2 \in (0, 1)$ is a tunable parameter. This means that a new map is created if the quantization error of unit i is too high when compared to that of the root unit. The higher the parameter τ_2, the fewer maps are created, and *vice versa*.

The new maps are created with a minimal topology of 2×2 units. In order to initialize each new unit, we compute the mean of the prototypes of the three immediate neighbors of the parent unit in the parent map which point to the direction of the new unit (see Fig. 3). Then we initialize the prototype of the new unit as the average of this vector with the prototype of the parent unit. Let us assume that the parent unit i has lattice coordinates (q, r); then the prototypes of the 2×2 new units $\boldsymbol{\omega}_{s,t}$ are given by:

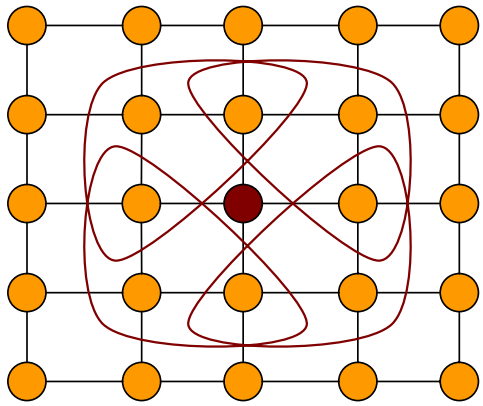


Fig. 3. Map creation. The parent unit is plotted in a darker tone. Each of the four closed curves encloses a set of three immediate neighbors of the parent unit.

$$\begin{aligned}\boldsymbol{\omega}_{1,1} = \frac{1}{2}\Bigg(\mathbf{w}_{q,r} + \frac{1}{3}(&\mathbf{w}_{q-1,r-1} \\ &+ \mathbf{w}_{q,r-1} + \mathbf{w}_{q-1,r})\Bigg),\end{aligned} \tag{34}$$

$$\begin{aligned}\boldsymbol{\omega}_{1,2} = \frac{1}{2}\Bigg(\mathbf{w}_{q,r} + \frac{1}{3}(&\mathbf{w}_{q-1,r+1} \\ &+ \mathbf{w}_{q,r+1} + \mathbf{w}_{q-1,r})\Bigg),\end{aligned} \tag{35}$$

$$\begin{aligned}\boldsymbol{\omega}_{2,1} = \frac{1}{2}\Bigg(\mathbf{w}_{q,r} + \frac{1}{3}(&\mathbf{w}_{q+1,r-1} \\ &+ \mathbf{w}_{q,r-1} + \mathbf{w}_{q+1,r})\Bigg),\end{aligned} \tag{36}$$

$$\begin{aligned}\boldsymbol{\omega}_{2,2} = \frac{1}{2}\Bigg(\mathbf{w}_{q,r} + \frac{1}{3}(&\mathbf{w}_{q+1,r+1} \\ &+ \mathbf{w}_{q,r+1} + \mathbf{w}_{q+1,r})\Bigg).\end{aligned} \tag{37}$$

If the parent unit does not have three immediate neighbors in some direction, we simply initialize the associated new prototype to the prototype of the parent unit $\mathbf{w}_{q,r}$. The effect of this initialization is that the units of the child map represent weighted combinations of the clusters of the neighbors of the parent unit. It must be noted that the training set for the child map is the set of input samples which belong to the cluster $\mathcal{C}_i$ of the parent unit.

2.5. *Learning algorithm*

The learning algorithm of the GHBSOM can be summarized as follows:

(1) The single unit of the root map is initialized to the overall mean of the input dataset.
(2) For each unit i which has not been tested for expansion yet, if (33) holds then we create a child map from it with 2×2 new units initialized by (34)–(37). Next we assign the training set of the new map to the parent cluster $\mathcal{C}_i$.
(3) For each map which has not been trained yet, we use (20) and (23) to adapt it to its training set. Then we enlarge the map if (27) holds, and we train the map again until (27) does not hold.
(4) If there is any unit which has not been tested for expansion, then go to step 2. Otherwise, halt.

2.6. *Probabilistic interpretation*

The neurons of the GHBSOM model has been characterized in (3) as minimizers of the divergence of their associated training samples. This is in line with the classic interpretation of the SOM and the GHSOM as minimizers of the sum of the squared Euclidean errors of the training data. However, the association of Bregman divergences to probability density functions given by (6) enables a probabilistic interpretation of the GHBSOM that is outlined next.

Let us consider a map with N neurons in a GHBSOM hierarchy with divergence D_ϕ. Then the following probabilistic mixture model is associated to the map, where the ith mixture component corresponds to the ith neuron of the map:

$$p(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} p(\mathbf{x} \,|\, i). \tag{38}$$

Equal *a priori* probabilities are assumed, as commonly done in probabilistic SOMs.[33] In case that neuron i has a child map, then $p(\mathbf{x} \,|\, i)$ is recursively defined by (38) as the probability density associated to the child map. Otherwise, the probability density function associated to a childless neuron i is given by:

$$p(\mathbf{x} \,|\, i) = \exp(-D_\phi(\mathbf{x}, \mathbf{w}_i))\beta_\phi(\mathbf{x}), \tag{39}$$

which comes from (6).

In this context, the computation of the winning neuron (20) can be reformulated as the computation of the neuron which has the highest likelihood to have generated the observed training sample $\mathbf{x}(n)$ at time-step n:

$$\begin{aligned} \text{Winner}(n) &= \arg \max_{i \in \{1,\ldots,N\}} P(i \,|\, \mathbf{x}(n)) \\ &= \arg \max_{i \in \{1,\ldots,N\}} p(\mathbf{x}(n) \,|\, i), \end{aligned} \tag{40}$$

where it must be highlighted that:

$$\begin{aligned} &\arg \max_{i \in \{1,\ldots,N\}} p(\mathbf{x}(n) \,|\, i) \\ &\quad = \arg \max_{i \in \{1,\ldots,N\}} \exp(-D_\phi(\mathbf{x}(n), \mathbf{w}_i(n))) \\ &\quad \arg \min_{i \in \{1,\ldots,N\}} D_\phi(\mathbf{x}(n), \mathbf{w}_i(n)), \end{aligned} \tag{41}$$

since $\beta_\phi(\mathbf{x}(n))$ does not depend on the prototype $\mathbf{w}_i(n)$, as noted in Sec. 2.1.

2.7. *Performance comparison*

The fact that a GHBSOM network might use any Bregman divergence poses the fundamental problem of how to compare several networks to choose the best performing one. Each Bregman divergence defines a different quantization error criterion (24), so it is not possible to use the quantization error as a performance criterion. There are some applications where performance measures are available which are not associated to a particular definition of the quantization error. For example, for classification we could consider the classification accuracy or the Rand index; and for unsupervised clustering we could consider the Mean Silhouette Value. For these applications the above mentioned problem does not arise, since those application specific performance measures can be used. For all the other cases, it is advantageous to have a problem independent performance measure which can be used to compare GHBSOM networks with different divergences.

Since the GHBSOM is fundamentally a self-organizing system, it is natural to measure how well the network has self-organized. The most popular self-organization performance measure is the topographic error, both for the evaluation of new map models and learning algorithms[34–37] and for the development of practical applications.[38–40] For a flat SOM, i.e. non-hierarchical, it is defined as the fraction of test samples whose best matching unit is not a topological neighbor of the second best matching

unit:

$$\mathrm{TE}_{\mathrm{SOM}} = \frac{1}{K}\sum_{k=1}^{K}\mathbb{I}(d(\mathrm{Winner}(\mathbf{x}(k)), \mathrm{Second}(\mathbf{x}(k))) > 1), \quad (42)$$

$$\mathrm{Second}(\mathbf{x}(k)) = \arg\min_{j\in\{1,\ldots,M\}, j\neq\mathrm{Winner}(\mathbf{x}(k))} \times D_\phi(\mathbf{x}(k), \mathbf{w}_i(k)). \quad (43)$$

Please note that the topographic error lies in the closed interval $[0, 1]$ (lower is better). A high value of $\mathrm{TE}_{\mathrm{SOM}}$ means that the computational map is not well ordered. The definition (42) cannot be applied directly to a GHBSOM because it does not consider any hierarchy. The following equation is proposed as the topographic error measure for a GHBSOM:

$$\mathrm{TE}_{\mathrm{GHBSOM}} = \frac{1}{K}\sum_{k=1}^{K}\tau_{\mathcal{M}_0}(\mathbf{x}(k)), \quad (44)$$

where $\mathcal{M}_0$ is the child map of the root unit of the GHBSOM hierarchy. The function $\tau_{\mathcal{M}}$ is recursively defined for a map $\mathcal{M}$ as follows. If the winning neuron in $\mathcal{M}$ for the test sample $\mathbf{x}(k)$ is childless, then the result is $\mathbb{I}(d_{\mathcal{M}}(\mathrm{Winner}(\mathbf{x}(k)), \mathrm{Second}(\mathbf{x}(k))) > 1)$. Otherwise, the result is $\tau_{\mathrm{Child}(\mathrm{Winner}(\mathbf{x}(k)))}(\mathbf{x}(k))$. Please note that for a GHBSOM with no children below $\mathcal{M}_0$, the proposed measure matches the classic one: $\mathrm{TE}_{\mathrm{GHBSOM}} = \mathrm{TE}_{\mathrm{SOM}}$.

There is still another difficulty, namely the possibility that the number of neurons of the maps are different. This poses a problem because it is more likely to have lower values of the topographic error for smaller maps, since the fraction of topological neighbors is higher. Let $\bar{\mathrm{TE}}_{\mathrm{GHBSOM}}$ be the expected value of $\mathrm{TE}_{\mathrm{GHBSOM}}$ for a GHBSOM with the same structure as the network at hand, but randomly chosen prototypes:

$$\bar{\mathrm{TE}}_{\mathrm{GHBSOM}} = \bar{\tau}_{\mathcal{M}_0}, \quad (45)$$

$$\bar{\tau}_{\mathcal{M}} = \frac{1}{N_{\mathcal{M}}}\left(\sum_{i\in\mathcal{L}(\mathcal{M})}\frac{\gamma_i}{N_{\mathcal{M}}-1} + \sum_{i\notin\mathcal{L}(\mathcal{M})}\bar{\tau}_{\mathrm{Child}(i)}\right), \quad (46)$$

where $N_{\mathcal{M}}$ is the number of neurons in map $\mathcal{M}$, $\mathcal{L}(\mathcal{M})$ is the set of childless neurons in map $\mathcal{M}$, and γ_i is the number of neurons which are not neighbors of neuron i in map $\mathcal{M}$.

Now a normalized version of $\mathrm{TE}_{\mathrm{GHBSOM}}$ can be obtained by subtracting the expected value and dividing by the difference between the maximum value and the expected value, as considered in Refs. 41 and 42:

$$\mathrm{NTE}_{\mathrm{GHBSOM}} = \frac{\mathrm{TE}_{\mathrm{GHBSOM}} - \bar{\mathrm{TE}}_{\mathrm{GHBSOM}}}{1 - \bar{\mathrm{TE}}_{\mathrm{GHBSOM}}}, \quad (47)$$

so that the GHBSOM network with the lowest value of $\mathrm{NTE}_{\mathrm{GHBSOM}}$ is preferred because it exhibits a lower topographic error. It must be highlighted that negative values of $\mathrm{NTE}_{\mathrm{GHBSOM}}$ are to be expected in all cases, since they indicate that the topographic error is better than that of a random GHBSOM.

3. Discussion

In this section, we discuss some important properties of our proposal:

(1) In this work, we have chosen Bregman divergences to develop a new growing hierarchical self-organizing network. The reason for this choice, among other dissimilarity measures which have been proposed for SOMs,[20] can be traced to Eq. (2). Bregman divergences are the only ones whose minimizer is the mean, and this makes them particularly suitable to growing hierarchical networks. This is because the initialization of new units is straightforward when the clusters are represented by the estimations of their means, given the linearity of the expectation operator. That is, we can simply set the new prototypes to suitable linear combinations of the old prototypes, as done in Secs. 2.3 and 2.4. This would be impossible for other dissimilarities, such as Csiszár's f-divergences or γ-divergences.

(2) A fundamental advantage of the stochastic approximation framework that has been chosen to develop the learning algorithm is that the learning equations have the same form for all Bregman divergences. Only the winning neuron computation (20) changes, and the modification amounts to using one divergence D_ϕ or another from those listed in the last column of Table 1. This constrasts with the stochastic gradient descent approach in Ref. 20, where specific equations must be derived for each divergence. Expectation maximization has also been used for SOMs, but it is mainly oriented to batch

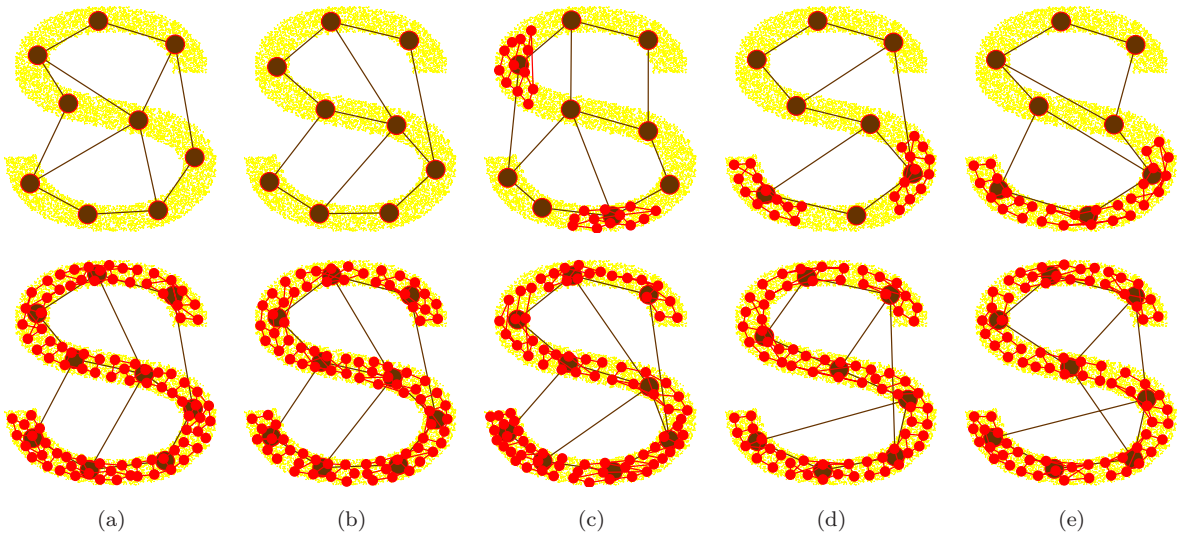
mode procesing,[33,43] while our proposal is oriented to online learning so that there is no need to provide the entire training set before the learning starts.
(3) The learning algorithm is based on the minimization of the Bregman divergences among the training data and the prototypes, as seen in Secs. 2.2–2.4. This is also the case of the standard SOM and GHSOM, with the only difference that these two models minimize the Euclidean distances rather than general Bregman divergences. For each specific application (clustering, classification, etc.) there are specific objective functions which are measured on the already trained networks, so that the network that optimizes the objective function is chosen. That is, the learning algorithm does not optimize the objective function of the application directly. Please note that this is the case for all SOM models, so the GHBSOM is fully in line with them in this sense.
(4) Care must be taken with the domain $\mathcal{S}$ of the generating function ϕ. It is always possible to shift the input dataset so that the vector components are positive, as required in most cases in Table 1. However, this poses the problem of the interpretation of a clustering over the shifted data. For a particular application, such a clustering might be meaningless if the differences $x_k - y_k$ used by Euclidean distance are more important than the ratios $\frac{x_k}{y_k}$ considered by other divergences. Hence we should not use Bregman divergences as black boxes, as is the case with any other data analysis technique. As a rule of thumb, we can say that if all the variables in the dataset are ratio variables, i.e. if the ratios $\frac{x_k}{y_k}$ are meaningful for all k, then we can use any of the five Bregman divergences listed in Table 1. Otherwise we should restrict our attention to squared Euclidean and exponential loss, since they do not depend on these ratios.
(5) Our model contains the original GHSOM,[15] as a particular case, when the chosen divergence is the squared Euclidean distance. Consequently, the comparisons in the experimental section are aimed to assess the relative advantage of using the non-Euclidean Bregman divergences instead of the Euclidean distance of the GHSOM.

4. Experiments

In this section, several experiments have been designed in order to show the performance of the proposed model. All the Bregman divergences considered in Table 1 are taken into account. First, the self-organization capabilities of the GHBSOM are shown (Sec. 4.1); then several datasets have been chosen to assess the classification performance of our model, as a typical supervised learning application (Sec. 4.2); third, the same datasets used for classification have been utilized to evaluate the clustering performance of our approach, as a unsupervised learning application (Sec. 4.3); and finally the GHBSOM has been applied for clustering and visualization of regions with the same weather conditions from satellite images as another unsupervised learning application (Sec. 4.4).

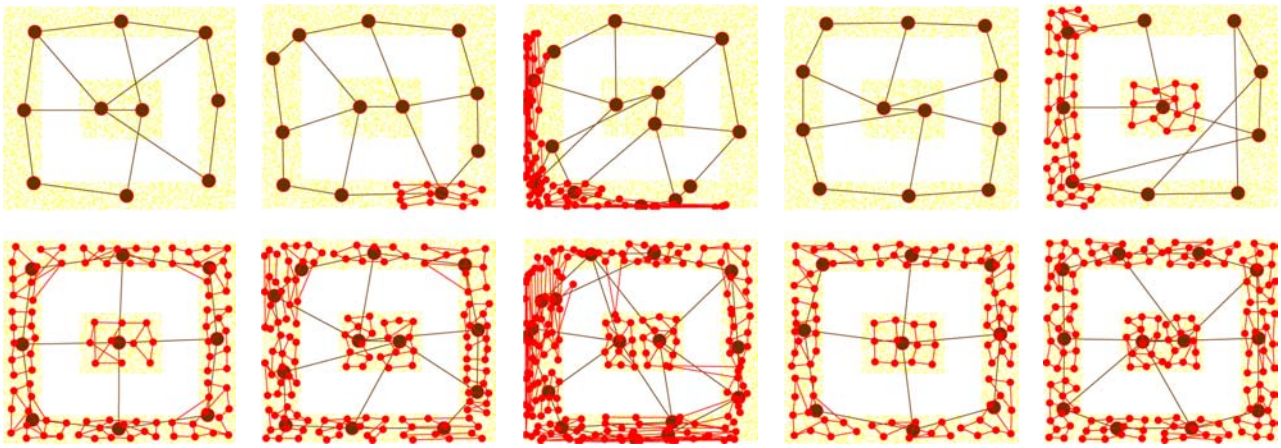
4.1. *Self organization experiments*

The first set of experiments is designed to show the self-organization capabilities of the GHBSOM for different Bregman divergences. Two-dimensional input distributions ($D = 2$) of 10,000 input samples were chosen. The input distributions are uniform distributions on a region with the shape of the "S" letter and a filled square. The training was done during 60,000 time-steps, with $\tau_1 = 0.01$ and two different values for the τ_2 parameter, $\tau_2 = 0.01$ and $\tau_2 = 0.001$, to see how the neural expansion process increases as τ_2 diminishes. The two input distributions and the resulting models for each divergence are shown in Figs. 4 and 5, where the models were obtained with $\tau_2 = 0.01$ in the first row and with $\tau_2 = 0.001$ in the second. In the plots, neurons are represented by circles whose size and tone depends on the layer of the hierarchy they belong to. Connections between neurons are plotted with straight lines with the same tone than the neurons they connect to. We can see how the models fit appropriately to the shapes for all the considered Bregman divergences. The input samples have also been plotted with small points. For the "S" letter distribution, by setting $\tau_1 = 0.01$ 8 or 9 neurons are obtained at the first layer and around 12 neurons at the second layer for all divergences. As seen, with a lower value of the τ_2 parameter more neurons are expanded into a new map at the next layer of the hierarchy, whereas for a higher value just two or three neurons are expanded for the three last divergences. A similar behavior can be noticed for the "filled square" distribution.



(a) (b) (c) (d) (e)

Fig. 4. GHBSOM results for the two-dimensional "S" letter distribution with $\tau_1 = 0.01$, $\tau_2 = 0.01$ (first row) and $\tau_2 = 0.001$ (second row): (a) squared Euclidean, (b) generalized I-divergence, (c) Itakura–Saito, (d) exponential loss and (e) logistic loss.



(a) (b) (c) (d) (e)

Fig. 5. GHBSOM results for the two-dimensional "filled square" distribution with $\tau_1 = 0.01$, $\tau_2 = 0.01$ (first row) and $\tau_2 = 0.001$ (second row): (a) squared Euclidean, (b) generalized I-divergence, (c) Itakura–Saito, (d) exponential loss and (e) logistic loss.

The normalized topographic error NTE has been computed over 10 runs for each of the dataset and parameter choices considered above. The results are given in Table 2. As explained in Sec. 2.7, the NTE is proposed as a performance measure which allows to compare GHBSOM models corresponding to different Bregman divergences and hierarchical structures. As seen, the logistic loss divergence attains the best results in terms of topology preservation, which means that it could be chosen over the others in case that no problem specific performance measures were available. It is also worth noting that all the values in Table 2 are clearly negative, i.e. the topology preservation of the trained GHBSOMs is much better than that of a model with random prototypes, which would yield NTE = 0 on average.

In order to analyze the effect of the GHBSOM learning parameters τ_1 and τ_2 in the results, an

Table 2. Normalized topographic error of the GHBSOM (lower is better) for the "S" and "filled square" datasets using five different Bregman divergences, with $\tau_1 = 0.01$, $\tau_2 = 0.01$ (first and second row) and $\tau_2 = 0.001$ (third and fourth rows). Standard deviations are shown in parentheses. Best results are highlighted in bold for each dataset and parameter choice (row).

	Squared Euclidean	Generalized I-divergence	Itakura–Saito	Exponential loss	Logistic loss
"S", $\tau_2 = 0.01$	−1.69 (0.19)	−1.66 (0.16)	−1.71 (0.68)	−1.66 (0.29)	**−1.72 (0.24)**
"Filled square", $\tau_2 = 0.01$	−1.99 (0.60)	−2.33 (0.58)	−2.42 (0.27)	−2.12 (0.46)	**−2.75 (0.47)**
"S", $\tau_2 = 0.001$	−1.53 (0.21)	−1.53 (0.21)	−1.51 (0.21)	−1.46 (0.15)	**−1.75 (0.12)**
"Filled square", $\tau_2 = 0.001$	−1.85 (0.17)	−1.78 (0.14)	−1.57 (0.13)	−1.80 (0.11)	**−2.19 (0.15)**

experiment with the "S" letter distribution has been carried out. For each combination of parameter values with $\tau_1, \tau_2 \in \{2^h \cdot 10^{-3}, h = 0, \ldots, 9\}$, 100 simulations have been run. The adaptation to the input distribution has been measured by the mean quantization error (MQE), which is computed by recursive application of (24) in the GHBSOM hierarchy, i.e. for each test sample $\mathbf{x}$ the minimum quantization error over the leaf neurons is considered. In order to assess the complexity and shape of the

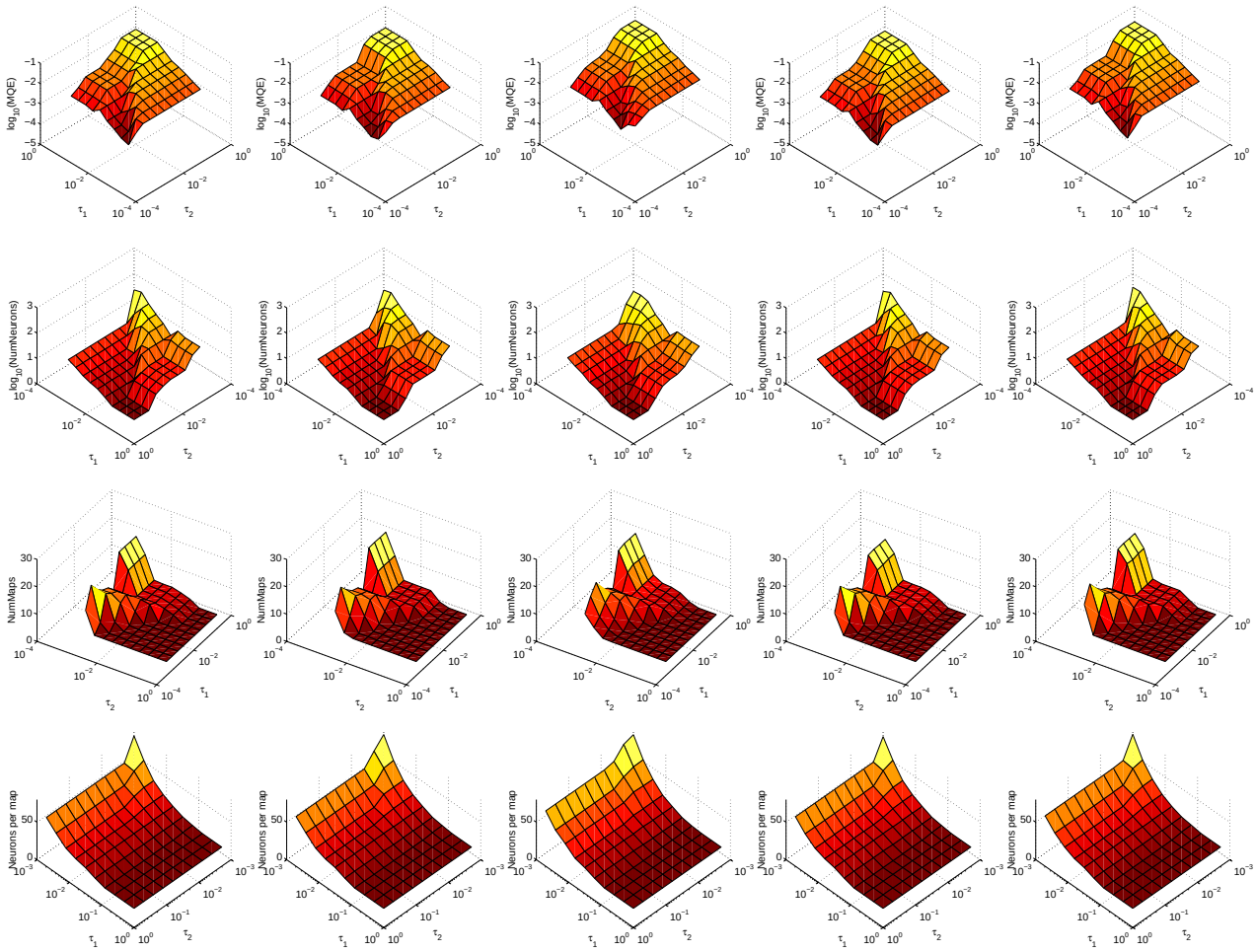


Fig. 6. Effect of the τ_1 and τ_2 learning parameters. From left to right: squared Euclidean, generalized I-divergence, Itakura–Saito, exponential loss and logistic loss. From top to bottom: mean quantization error, overall number of neurons, number of maps and number of neurons per map. Lower values are marked with darker tones.

learned hierarchy, three quantitative measures have been computed: the overall number of neurons, the number of maps and the number of neurons per map.

The results are depicted in Fig. 6; for each combination of values of τ_1 and τ_2, the average of each quantitative measure over the 100 simulations is shown. It can be noticed that lower quantization errors (first row) are associated with higher overall numbers of neurons (second row), as could be expected since more neurons can represent the input distribution in more detail. It can be seen in the third row that the number of maps is strongly dependent on the value of τ_2, which determines when a new map is created (see Sec. 2.4). The lower τ_2, the more maps are created and *vice versa*. Finally, the fourth row shows that the number of neurons of the maps is highly dependent on the value of τ_1, which controls whether a map must grow (see Sec. 2.3). The lower τ_1, the more neurons are added to the maps and *vice versa*. The combined effect of τ_1 and τ_2 is that small GHBSOMs are obtained for high values of both parameters, while large models are built when both parameters are small. As mentioned before, large GHBSOMs are associated with low values of MQE. A final conclusion is that the effect of τ_1 and τ_2 is the same for all the considered Bregman divergences; please note that the five columns look the same.

4.2. *Classification of UCI datasets*

This set of experiments is devoted to show the classification performance of the GHBSOM for the considered Bregman divergences (see Sec. 1). Classification is a fundamental task in machine learning applications,[44–46] and it has been widely studied in the field of self-organizing neural networks.[29,47–51] There are different ways to use self-organizing maps for classification.[33] Here, the "one map per class" strategy has been selected. A GHBSOM network $\mathcal{N}_j$ is trained with the available training samples of each class S_j. Then the Bayesian class probabilities of a test sample $\mathbf{x}$ are computed with the help of the probability density functions $p_{\mathcal{N}_j}$ associated by Eq. (38) to each network $\mathcal{N}_j$:

$$P(S_j \mid \mathbf{x}) = \frac{P(S_j) p_{\mathcal{N}_j}(\mathbf{x})}{\sum_h P(S_h) p_{\mathcal{N}_h}(\mathbf{x})}. \tag{48}$$

The above equation assumes that the map probability density function models the class density function adequately:

$$p(\mathbf{x} \mid S_j) = p_{\mathcal{N}_j}(\mathbf{x}) \tag{49}$$

and this calls for the use of a Bregman divergence whose associated probability density fits the dataset at hand. Accuracy and other standard classification performance measures can be employed to carry out this selection, as discussed in Sec. 3.

For our experiments, eight benchmark datasets from the UCI Machine Learning Repository have been selected.[52] Each dataset has been normalized by shifting and rescaling them to lie in the interval [0,1]. Four configurations of the τ_1 and τ_2 parameters have been used: $\tau_1 = 0.1$ and $\tau_2 = 0.01$; $\tau_1 = 0.01$ and $\tau_2 = 0.1$; $\tau_1 = 0.01$ and $\tau_2 = 0.01$; and $\tau_1 = 0.01$ and $\tau_2 = 0.001$. For each dataset and configuration, 10 folds have been run, randomly dividing the data in 90% for training and 10% for test. The training was done during two epochs.

Table 3. Classification accuracy of the GHBSOM for several datasets using five different Bregman divergences (higher is better). The best results from different configurations of the τ_1 and τ_2 parameters were selected. Standard deviations are shown in parentheses. Best results are highlighted in bold for each dataset (row).

	Squared Euclidean	Generalized I-divergence	Itakura–Saito	Exponential loss	Logistic loss
Balance scale	0.89 (0.02)	**0.91 (0.04)**	0.87 (0.05)	0.88 (0.05)	0.90 (0.05)
Breast Cancer Wisconsin	0.94 (0.03)	**0.97 (0.02)**	0.96 (0.02)	0.88 (0.04)	0.96 (0.02)
Cloud	0.95 (0.03)	0.96 (0.01)	**1.00 (0.01)**	0.93 (0.03)	0.98 (0.01)
Contraceptive	**0.46 (0.04)**	0.46 (0.04)	0.43 (0.06)	0.44 (0.04)	0.46 (0.03)
Dermatology	0.94 (0.03)	**0.98 (0.02)**	0.94 (0.03)	0.93 (0.04)	0.96 (0.02)
Liver	0.64 (0.09)	0.64 (0.08)	**0.73 (0.08)**	0.61 (0.08)	0.59 (0.07)
Vowel	0.36 (0.05)	0.34 (0.06)	0.45 (0.05)	0.29 (0.05)	**0.48 (0.05)**
Wine	0.64 (0.17)	0.72 (0.11)	**0.85 (0.11)**	0.57 (0.10)	0.83 (0.09)

Table 4. Rand index of the GHBSOM for several datasets using five different Bregman divergences (higher is better). The best results from different configurations of the τ_1 and τ_2 parameters were selected. Standard deviations are shown in parentheses. Best results are highlighted in bold for each dataset (row).

	Squared Euclidean	Generalized I-divergence	Itakura–Saito	Exponential loss	Logistic loss
Balance scale	0.86 (0.02)	**0.91 (0.04)**	0.84 (0.06)	0.85 (0.07)	0.90 (0.06)
Breast Cancer Wisconsin	0.89 (0.05)	**0.95 (0.04)**	0.92 (0.03)	0.78 (0.06)	0.93 (0.04)
Cloud	0.90 (0.05)	0.92 (0.02)	**0.99 (0.01)**	0.87 (0.05)	0.96 (0.02)
Contraceptive	0.44 (0.05)	0.49 (0.01)	**0.56 (0.02)**	0.45 (0.06)	0.53 (0.02)
Dermatology	0.97 (0.01)	**0.98 (0.02)**	0.95 (0.03)	0.96 (0.02)	0.98 (0.01)
Liver	0.54 (0.05)	0.54 (0.06)	**0.61 (0.08)**	0.52 (0.03)	0.51 (0.03)
Vowel	0.84 (0.01)	0.83 (0.03)	0.86 (0.01)	0.81 (0.03)	**0.87 (0.01)**
Wine	0.59 (0.15)	0.71 (0.12)	**0.82 (0.13)**	0.48 (0.09)	0.79 (0.11)

Table 5. Overall entropy of the GHBSOM for several datasets using five different Bregman divergences (lower is better). The best results from different configurations of the τ_1 and τ_2 parameters were selected. Standard deviations are shown in parentheses. Best results are highlighted in bold for each dataset (row).

	Squared Euclidean	Generalized I-divergence	Itakura–Saito	Exponential loss	Logistic loss
Balance scale	0.29 (0.05)	**0.19 (0.07)**	0.31 (0.11)	0.31 (0.13)	0.21 (0.13)
Breast Cancer Wisconsin	0.18 (0.08)	**0.10 (0.07)**	0.16 (0.05)	0.30 (0.05)	0.14 (0.08)
Cloud	0.18 (0.08)	0.16 (0.04)	**0.02 (0.03)**	0.21 (0.05)	0.08 (0.04)
Contraceptive	**0.72 (0.11)**	0.83 (0.03)	1.00 (0.02)	0.74 (0.13)	0.95 (0.04)
Dermatology	0.14 (0.06)	**0.07 (0.07)**	0.16 (0.08)	0.14 (0.09)	0.09 (0.06)
Liver	**0.32 (0.02)**	0.34 (0.04)	0.50 (0.10)	0.33 (0.02)	0.38 (0.05)
Vowel	1.31 (0.09)	1.34 (0.08)	1.23 (0.12)	1.35 (0.10)	**1.15 (0.09)**
Wine	0.45 (0.14)	0.34 (0.15)	**0.31 (0.20)**	0.51 (0.09)	0.35 (0.17)

In order to assess the classification performance of the resulting GHBSOM models, three quality measures have been taken into account from those methods proposed by Moschou *et al.*,[53] namely, classification accuracy, Rand index and overall entropy. The best results from the different configurations of the τ_1 and τ_2 parameters are given in Tables 3–5. Note that the squared Euclidean divergence (used in the original Kohonen's SOM) shows one of the worst classification performances with regard to the other divergences. In order to see clearer the best divergence for classification, a classification ranking has been provided. Thus, for each dataset and configuration of the τ_1 and τ_2 parameters, a number between 1 and 5 (the number of considered divergences) has been assigned to each divergence after obtaining its classification accuracy, where 1 stands for the divergence with the best results and 5 the worst divergence. The classification performance ranking is given in Table 6. Here, we can see how the best divergences are the logistic loss and the I-divergence whereas the exponential loss and the Euclidean distance yield the worst results.

Table 6. Ranking of the classification performance of the GHBSOM for several datasets using five different Bregman divergences (lower is better). Best result is highlighted in bold.

Divergence	Ranking
Logistic loss	**18**
Generalized I-divergence	**18**
Itakura–Saito	22
Squared Euclidean	25
Exponential loss	37

4.3. *Clustering of UCI datasets*

This third set of experiments has been designed to assess the clustering abilities of the GHBSOM, as

a typical unsupervised learning task. For clustering applications, the availability of different Bregman divergences allows to form clusters of different shapes. This improves the capability of the GHBSOM to adapt to datasets with different cluster structures. If we consider two different divergences ϕ and ϕ' with the same set of centroids $\boldsymbol{\mu}_j$, the associated clusters are:

$$\mathcal{C}_i = \left\{ \mathbf{x} \in \mathcal{S} \,|\, i = \arg \min_{j \in \{1,\ldots,N\}} D_\phi(\mathbf{x}, \boldsymbol{\mu}_j) \right\}, \quad (50)$$

$$\mathcal{C}'_i = \left\{ \mathbf{x} \in \mathcal{S} \,|\, i = \arg \min_{j \in \{1,\ldots,N\}} D_{\phi'}(\mathbf{x}, \boldsymbol{\mu}_j) \right\}, \quad (51)$$

respectively. In general we will have $\mathcal{C}_i \neq \mathcal{C}'_i$ because of their different shapes, even if their means are the same:

$$\boldsymbol{\mu}_i = E[\mathbf{x} \,|\, \mathcal{C}_i] = E[\mathbf{x} \,|\, \mathcal{C}'_i]. \quad (52)$$

For these experiments we have utilized the same eight benchmark datasets from the UCI Machine Learning Repository[52] as used in the previous Sec. 4.2. The same four configurations of the τ_1 and τ_2 parameters used for classification have been selected: $\tau_1 = 0.1$ and $\tau_2 = 0.01$; $\tau_1 = 0.01$ and $\tau_2 = 0.1$; $\tau_1 = 0.01$ and $\tau_2 = 0.01$; and $\tau_1 = 0.01$ and $\tau_2 = 0.001$. For each dataset and configuration, 10 folds have been run during two epochs, randomly dividing the data in 90% for training and 10% for test. Each dataset has been normalized between 0 and 1.

In order to evaluate the clustering performance of the GHBSOM, silhouettes have been computed for each cluster. Silhouettes values indicate which data lie well within their clusters, and which ones are merely somewhere in between clusters.[54] A silhouette value is a number between -1 and 1 (higher is better). For each trained model, we computed the mean and the standard deviation of the silhouette values of the samples as a measure of how appropriately the data have been clustered. The best results from the different configurations of τ_1 and τ_2 parameters are given in Table 7. Please, note that the best clusterings have been performed by including Bregman divergences different from the squared Euclidean distance.

4.4. *Clustering of multispectral data*

The last experiment consists on performing an unsupervised clustering task of multispectral data and then visualizing the resulting clustering, which is a typical application of unsupervised neural networks.[55] In fact, some SOM models are specifically tailored to suit the needs of visualization applications. One of the best known ones is the Visualization-induced SOM (ViSOM[56]), which introduces a learning rule which regularizes the distance among the winning neuron and its neighbors so that the interneuron distances over the map grid are as homogeneous as possible. A generalized version (GViSOM[57]) is also able to manage mixed data with numerical and categorical variables. In addition to this, the results of an ensemble of ViSOMs can be summarized in a coherent way with the approach considered in Ref. 37.

Our dataset was created from images of NASA satellites that monitor the world climate every

Table 7. Mean silhouette values of the GHBSOM for several datasets using five different Bregman divergences (higher is better). The best results from different configurations of the τ_1 and τ_2 parameters were selected. Standard deviations are shown in parentheses. Best results are highlighted in bold for each dataset (row).

	Squared Euclidean	Generalized I-divergence	Itakura–Saito	Exponential loss	Logistic loss
Balance scale	0.93 (0.05)	**0.95 (0.05)**	0.77 (0.06)	0.94 (0.04)	0.93 (0.06)
Breast Cancer Wisconsin	0.88 (0.08)	**0.91 (0.05)**	0.89 (0.05)	0.87 (0.07)	0.90 (0.05)
Cloud	0.53 (0.02)	0.50 (0.04)	**0.54 (0.05)**	0.53 (0.02)	0.50 (0.05)
Contraceptive	0.79 (0.06)	0.77 (0.04)	0.70 (0.06)	**0.81 (0.05)**	0.78 (0.03)
Dermatology	0.92 (0.05)	0.93 (0.05)	**0.95 (0.08)**	0.91 (0.07)	0.93 (0.05)
Liver	0.93 (0.10)	0.92 (0.10)	0.93 (0.05)	0.95 (0.06)	**0.96 (0.03)**
Vowel	0.90 (0.03)	**0.91 (0.03)**	0.91 (0.05)	0.90 (0.04)	0.90 (0.03)
Wine	**1.00 (0.00)**	0.99 (0.02)	0.99 (0.02)	**1.00 (0.00)**	0.96 (0.09)

month.[a] In particular, nine types of images were chosen, namely, Aerosol Optical Depth, Aerosol Size, Carbon Monoxide, Chlorophyll Concentration, Cloud Fraction, Land Surface Temperature, Net Radiation, Vegetation and Water Vapor. The images we obtained are from February, 2011. The size of the images was rescaled to 900×450 pixels and then, a dataset of 405,000 samples and nine features corresponding to each image type was built. With this dataset, the aim is to perform a clustering of regions with a similar climate.

The training was done during two epochs, running 10 folds for each considered Bregman divergence. The τ_1 and τ_2 parameters were set to 0.01 and 0.001 in order to achieve larger GHBSOM architectures to improve the visualization of the clustering performed. For each considered Bregman divergence, the GHBSOM model from the 10 folds with the least quantization error was selected. The same procedure was carried out for the ViSOM, which we have chosen as a reference model for SOM visualization. The obtained models were used to visualize the induced clustering of regions. These visualizations are shown in Fig. 7. In the plots, the regions with the same color are meant to have the same climate. We can see how for the squared Euclidean divergence, for example certain regions of Asia have the same color as Antarctica, which is incorrect. In addition, this

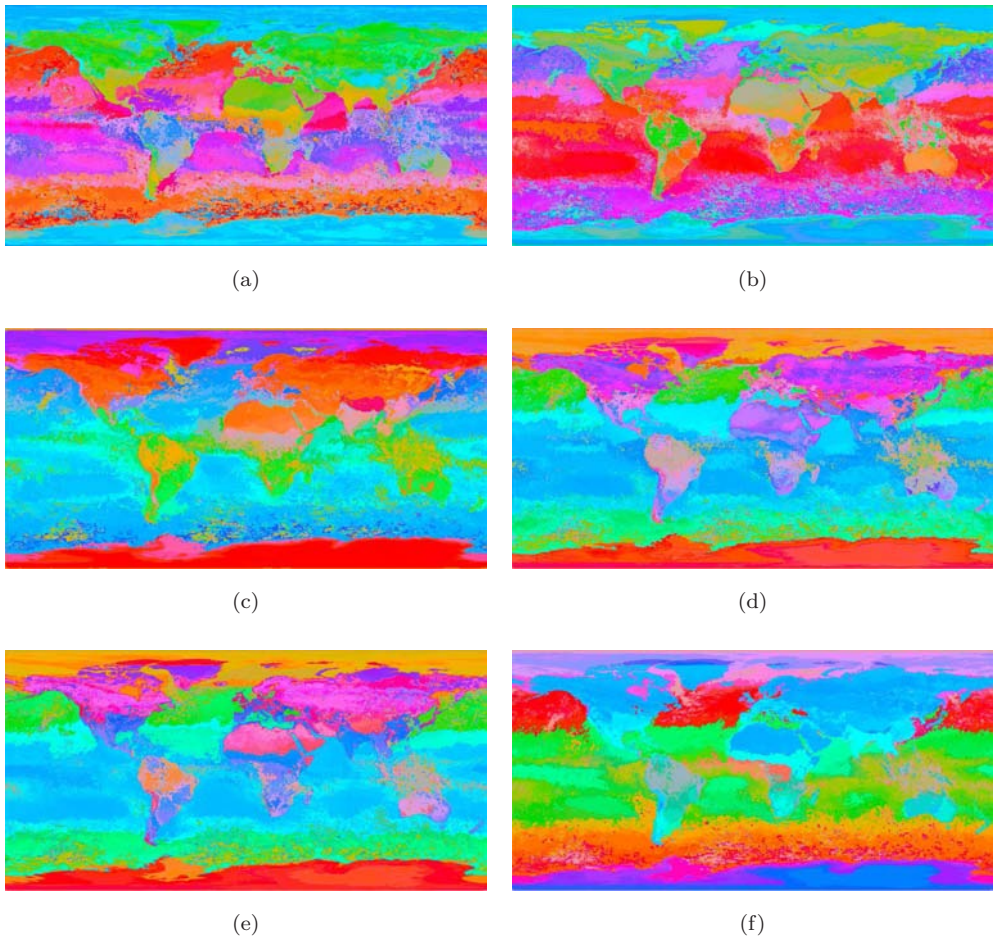
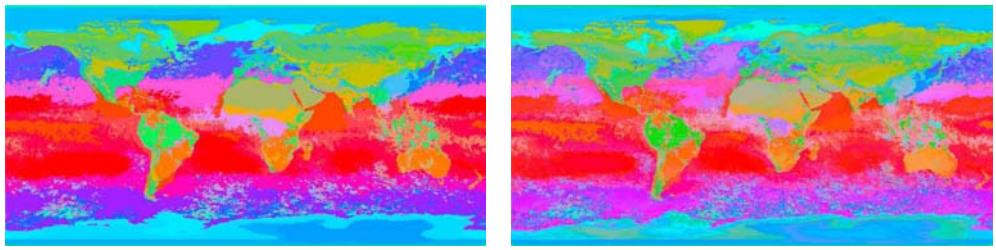


Fig. 7. Climate clustering results from satellite images. GHBSOM with $\tau_1 = 0.01$, $\tau_2 = 0.001$: (a) Squared Euclidean, (b) generalized I-divergence, (c) Itakura–Saito, (d) exponential loss and (e) logistic loss. ViSOM results are shown in (f).

[a]Available online: http://earthobservatory.nasa.gov/GlobalMaps/.



(a) (b)

Fig. 8. Climate clustering results of the GHBSOM with $\tau_1 = 0.01$, $\tau_2 = 0.001$ using the generalized I-divergence: (a) first layer and (b) first and second layer.

plot has a lot of noise, which incorrectly indicates that there are different weather conditions around the same area. A better representation is achieved for the rest of the considered divergences, where the regions with the same color are more coherent and have similar weather conditions. The ViSOM commits some mistakes, such as clustering Siberia and Sahara Desert together. For the generalized I-divergence, the obtained clustering is visualized for each of the two layers of the hierarchy in Fig. 8. Thus, we can see how certain regions from the first layer (e.g. the Atlantic Ocean) are represented at a higher level of detail in the second layer. Note that in the first layer plot, the color stripes are approximately symmetric from the Equator, as it could be expected. Non hierarchical models such as ViSOM are not able to visualize data this way, i.e. in increasing levels of detail.

5. Conclusions

In this paper, a new growing hierarchical self-organizing network based on Bregman divergences is proposed. This stands as a novel application of Bregman divergences, which have been successfully applied in the past to flat SOMs. The enhanced features of the new model improve the representation capability of the input data. Five Bregman divergences have been considered in this work, using several datasets to assess the classification performance of the proposed model and to perform clustering of multispectral data. According to the experiments, the generalized I-divergence achieves good performance in the classification and clustering of UCI datasets applications, whilst the squared Euclidean distance (the criterion used by the original Kohonen's SOFM, the GHSOM and many others) yields poor results. Moreover, the squared Euclidean makes some mistakes in the clustering of multispectral data experiments, whilst both the generalized I-divergence and the exponential loss get more coherent results. Furthermore, the self-organization experiments show that the Logistic Loss divergence attains the best results in terms of topology preservation.

In summary, experimental results demonstrate the improvement of self-organization ability for clustering applications introducing Bregman divergences in the growth process of hierarchical models. Furthermore, a deep analysis of the selected Bregman divergences has been carried out for classification applications, which shows that some Bregman divergences, such as the generalized I-divergence and the Logistic Loss divergences, outperform the squared Euclidean distance. This suggests that the introduction of Bregman divergences in other self-organizing networks could lead to similar improvements.

Acknowledgments

This work is partially supported by the Ministry of Economy and Competitiveness of Spain under project TIN2011-24141; and by the Autonomous Government of Andalusia (Spain) under projects TIC-6213 and TIC-657. All of them include FEDER funds. The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Malaga.

References

1. T. Kohonen, Self-organized formation of topologically correct feature maps, *Biol. Cybernet.* **43**(1) (1982) 59–69.
2. A. Fatehi and K. Abe, Flexible structure multiple modeling using irregular self-organizing maps neural network, *Int. J. Neural Syst.* **18**(3) (2008) 233–256.

3. E. López-Rubio, R. M. Luque-Baena and E. Domínguez, Foreground detection in video sequences with probabilistic self-organizing maps, *Int. J. Neural Syst.* **21**(3) (2011) 225–246.
4. C. Fyfe, W. Barbakh, W. C. Ooi and H. Ko, Topological mappings of video and audio data, *Int. J. Neural Syst.* **18**(6) (2008) 481–489.
5. N. Wang, M. J. Er, X. Meng and X. Li, An online self-organizing scheme for parsimonious and accurate fuzzy neural networks, *Int. J. Neural Syst.* **20**(5) (2010), 389–403.
6. E. Nichols, L. J. McDaid and M. N. H. Siddique, Case study on a self-organizing spiking neural network for robot navigation, *Int. J. Neural Syst.* **20**(6) (2010) 501–508.
7. J. Blackmore and R. Miikkulainen, Incremental grid growing: Encoding high-dimensional structure into a two-dimensional feature map, *IEEE Int. Conf. Neural Networks* (IEEE, 1993), pp. 450–455.
8. B. Fritzke, Growing cell structures — A self-organizing network for unsupervised and supervised learning, *Neural Networks* **7**(9) (1994) 1441–1460.
9. B. Fritzke, A Growing Neural Gas Network Learns Topologies, *Adv. Neural Inform. Process. Syst.* **7**(1) (1995) 625–632.
10. O. Beyer and P. Cimiano, Online semi-supervised growing neural gas, *Int. J. Neural Syst.* **22**(5) (2012) 1250023.
11. B. Fritzke, Growing grid — A self-organizing network with constant neighborhood range and adaptation strength, *Neural Process. Lett.* **2**(5) (1995) 9–13.
12. H. U. Bauer and T. Villmann, Growing a hypercubical output space in a self-organizing feature map, *IEEE Trans. Neural Networks* **8**(2) (1997) 218–226.
13. R. Mukkulainen, Script recognition with hierarchical feature maps, *Connection Sci.* **2**(1–2) (1990) 83–101.
14. J. Pakkanen, J. Iivarinen and E. Oja, The evolving tree, A novel self-organizing network for data analysis, *Neural Process. Lett.* **20**(3) (2004) 199–211.
15. A. Rauber, D. Merkl and M. Dittenbach, The growing hierarchical self-organizing map: Exploratory analysis of high-dimensional data, *IEEE Trans. Neural Networks* **13**(6) (2002) 1331–1341.
16. C. Hung and S. Wermter, A dynamic adaptive self-organising hybrid model for text clustering, *Third IEEE Int. Conf. Data Mining* (IEEE Comput. Soc., 2003), pp. 75–82.
17. M. Dittenbach, A. Rauber and G. Polzlbauer, Investigation of alternative strategies and quality measures for controlling the growth process of the growing hierarchical self-organizing map, *Proc. 2005 IEEE Int. Joint Conf. Neural Networks, 2005. IJCNN '05*, Vol. 5 (IEEE, 2005), pp. 2954–2959.
18. L. Bregman, The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming, *USSR Comput. Math. Math. Phys.* **7**(3) (1967) 200–217.
19. Y. Censor and S. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications* (Oxford University Press, 1998).
20. T. Villmann and S. Haase, Divergence-based vector quantization, *Neural Comput.* **23** (2011) 1343–1392.
21. A. Banerjee, S. Merugu, I. S. Dhillon and J. Ghosh, Clustering with Bregman divergences, *J. Mach. Learning Res.* **6** (2005) 1705–1749.
22. E. Jang, C. Fyfe and H. Ko, Bregman divergences and the self organising map, *Lecture Notes Comput. Sci.* **5326** (2008) 452–458.
23. E. Mwebaze, P. Schneider, F. M. Schleif, J. R. Aduwo, J. A. Quinn, S. Haase, T. Villmann and M. Biehl, Divergence-based classification in learning vector quantization, *Neurocomputing* **74**(9) (2011) 1429–1435.
24. H. J. Kushner and G. G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications* (Springer-Verlag, New York, NY, USA, 2003).
25. T. Lai, Stochastic approximation, *Ann. Statist.* **31**(2) (2003) 391–406.
26. B. Delyon, M. Lavielle and E. Moulines, Convergence of a stochastic approximation version of the EM algorithm, *Annals Statist.* **27**(1) (1999) 94–128.
27. M. Sato and S. Ishii, On-line EM algorithm for the normalized Gaussian network, *Neural Comput.* **12**(2) (2000) 407–432.
28. E. López-Rubio, J. M. Ortiz-de-Lazcano-Lobato and D. López-Rodríguez, Probabilistic PCA self-organizing maps, *IEEE Trans. Neural Networks* **20**(9) (2009) 1474–1489.
29. E. López-Rubio and E. Palomo, Growing hierarchical probabilistic self-organizing graphs, *IEEE Trans. Neural Networks* **22**(7) (2011) 997–1008.
30. E. López-Rubio, E. J. Palomo-Ferrer, J. M. Ortiz-de Lazcano-Lobato and M. C. Vargas-González, Dynamic topology learning with the probabilistic self-organizing graph, *Neurocomputing* **74**(16) (2011) 2633–2648.
31. H. Robbins and S. Monro, A stochastic approximation method, *Ann. Math. Statist.* **22**(3) (1951) 400–407.
32. T. Heskes, Self-organizing maps, vector quantization, and mixture modeling, *IEEE Trans. Neural Networks* **12**(6) (2001) 1299–1305.
33. E. López-Rubio, Probabilistic self-organizing maps for continuous data, *IEEE Trans. Neural Networks* **21**(10) (2010) 1543–1554.
34. A. Hsu and S. Halgamuge, Enhancement of topology preservation and hierarchical dynamic self-organising maps for data visualisation, *Int. J. Approx. Reason.* **32**(2–3) (2003) 259–279.
35. M. Merkow and R. K. DeLisle, Improving the performance of self-organizing maps via growing

representations, *J. Chem. Inform. Model.* **47**(5) (2007) 1797–1807.
36. B. Baruque and E. Corchado, A weighted voting summarization of SOM ensembles, *Data Mining Knowledge Disc.* **21**(3) (2010) 398–426.
37. E. Corchado and B. Baruque, WeVoS-ViSOM: An ensemble summarization algorithm for enhanced data visualization, *Neurocomputing* **75**(1) (2012) 171–184.
38. R. Céréghino and Y.-S. Park, Review of the self-organizing map (SOM) approach in water resources: Commentary, *Environ. Model. Software* **24**(8) (2009) 945–947.
39. K.-S. Jeong, D.-G. Hong, M.-S. Byeon, J.-C. Jeong, H.-G. Kim, D.-K. Kim and G.-J. Joo, Stream modification patterns in a river basin: Field survey and self-organizing map (SOM) application, *Ecol. Inform.* **5**(4) (2010) 293–303.
40. C.-W. Lee, J.-D. Jang, K.-S. Jeong, D.-K. Kim and G.-J. Joo, Patterning habitat preference of avifaunal assemblage on the Nakdong River estuary (South Korea) using self-organizing map, *Ecol. Inform.* **5**(2) (2010) 89–96.
41. L. Hubert and P. Arabie, Comparing partitions, *J. Classification* **2** (1985) 193–218.
42. R. Unnikrishnan, C. Pantofaru and M. Hebert, Toward objective evaluation of image segmentation algorithms, *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(6) (2007) 929–944.
43. H. Yin and N. Allinson, Self-organizing mixture networks for probability density estimation, *IEEE Trans. Neural Networks* **12**(2) (2001) 405–411.
44. G. Rodríguez-Bermudez, P. García-Laencina and J. Roca-Dorda, Efficient automatic selection and combination of EEG features in least squares classifiers for motor-imagery brain computer interfaces, *Int. J. Neural Syst.* **23**(4) (2013) 1350015.
45. D. Alvarez, R. Hornero, J. Marcos, N. Wessel, T. Penzel and F. del Campo, Assessment of feature selection and classification approaches to enhance information from overnight oximetry in the context of sleep apnea diagnosis, *Int. J. Neural Syst.* **23**(5) (2013) 1350020.
46. M. Cabrerizo, M. Ayala, M. Goryawala, P. Jayakar and M. Adjouadi, A new parametric feature descriptor for the classification of epileptic and control EEG records in pediatric population, *Int. J. Neural Syst.* **22**(2) (2012) 1250001–1250016.
47. D. Merkl, Text classification with self-organizing maps: Some lessons learned, *Neurocomputing* **21** (1–3) (1998) 61–77.
48. A. Astel, S. Tsakovski, P. Barbieri and V. Simeonov, Comparison of self-organizing maps classification approach with cluster and principal components analysis for large environmental data sets, *Water Res.* **41**(19) (2007) 4566–4578.
49. S.-S. Cheng, H.-C. Fu and H.-M. Wang, Model-based clustering by probabilistic self-organizing maps, *IEEE Trans. Neural Networks* **20**(5) (2009) 805–826.
50. M. Rahman, W. Pi Yang, T. Chow and S. Wu, A flexible multi-layer self-organizing map for generic processing of tree-structured data, *Pattern Recogn.* **40**(5) (2007) 1406–1424.
51. H. Zheng, G. Lefebvre and C. Laurent, Fast-learning adaptive-subspace self-organizing map: An application to saliency-based invariant image feature construction, *IEEE Trans. Neural Networks* **19**(5) (2008) 746–757.
52. A. Asuncion and D. Newman, UCI machine learning repository (2007).
53. V. Moschou, D. Ververidis and C. Kotropoulos, Assessment of self-organizing map variants for clustering with application to redistribution of emotional speech patterns, *Neurocomputing* **71**(1–3) (2007) 147–156.
54. P. J. Rousseeuw, Silhouettes: A graphical aid to the interpretation and validation of cluster analysis, *J. Comput. Appl. Math.* **20**(0) (1987) 53–65.
55. A. Herrero, U. Zurutuza and E. Corchado, A neural-visualization ids for honeynet data, *Int. J. Neural Syst.* **22**(2) (2012) 1250005–1250018.
56. H. Yin, ViSOM-a novel method for multivariate data projection and structure visualization, *IEEE Trans. Neural Networks* **13**(1) (2002) 237–243.
57. C.-C. Hsu, K.-M. Wang and S.-H. Wang, GViSOM for multivariate mixed data projection and structure visualization, *Int. Joint Conf. Neural Networks* (2006), pp. 3300–3305.