Self Organizing Map
A Probabilistic Approach

F. Anouar[1], F. Badran[1] et S.Thiria[1,2]

1 CEDRIC, Conservatoire National des Arts et Métiers, 292 rue Saint Martin, 75 003 PARIS
2 Laboratoire d'Océanographie et de Climatologie (LODYC), Université de PARIS 6. 4 Place Jussieu, 75005 PARIS
E-mail: anouar@cnam.fr

Abstract

The Self Organizing Map (SOM) of Kohonen has been successfully applied to wide class of problems. However, the first algorithm has been derived from heuristic ideas and not motivated by statistical considerations. In this paper we propose a new learning algorithm PRSOM (Probabilistic SOM) using a probabilistic formalism for topological maps. This algorithm approximates the density distribution of the inputs using a local mixture of normal distributions. The unsupervised learning is based on maximum of likelihood optimization, thus, contrary to SOM, the PRSOM algorithm optimizes a defined objective function. The paper present the PRSOM properties and derive the relationship between SOM and PRSOM. A supervised version of PRSOM using RBF (Radial Basis Function) network allows to prove the efficiency of the approach on real data.

1. Introduction

The Self Organizing Map of Kohonen is a vector quantization algorithm in which a set of unlabelled data vectors z_i (i=1..N) in a n-dimensional space is represented by a set of n-dimensional reference vectors which are linked according to a topological order. This order is defined using an undirected graph named the map whose cells are discrete points of a reduced D-dimensional space. Usually the D-dimension is much less than n in order to allow data visualization (D=1 or D=2). Kohonen algorithm is a non-linear projection of the discrete space of the cells into the n-dimensional data space in which the image of each cell corresponds to a reference vector. The main characteristic of the SOM algorithm is to preserve the topological order: two adjacent cells are mapped into two close reference vectors. Kohonen algorithm can be summarized as a vector quantization with a neighboring constraint. This constraint is taken into account through the learning rule which update simultaneously one unit together with the units within its predefined neighborhood. For a given observation z_i and its "winner" cell c_i, the usual "on-line" Kohonen algorithm uses the following learning rule:

$$W_c^t = W_c^{t-1} - \varepsilon(t)\, K_T(\delta(c, c_i))(W_c^{t-1} - z_i)$$

The kernel function K_T depends on the parameter T and allows to weight the influence of each cell c of the map C with respect to the winner cell c_i. Empirical simulations show that starting with large ε(t) and T and decreasing them as iterations progress gives the best qualitative results. This self-organizing process can be considered as a deterministic annealing [Martinez 93].

Usually the learning stage ends when $K_T(c, c_i) \approx 0$ for $c \neq c_i$ and the updating rule appears as being a k-means adaptive learning rule. In the following we denote by T* the value of the parameter T which restricts the modifications to the unique winner weight W_{c_i}. The SOM algorithm does not optimize an objective function, however, the convergence of the algorithm to a local minimum of a cost function (1) can be shown using the "batch" version of the Kohonen algorithm for a fixed value of T [Kohonen 93]:

$$E_T(\chi, W) = \sum_z \sum_{c \in C} K_T(\delta(c, \chi(z)) \left\| z - W_c \right\|^2 \qquad (1)$$

where C represents the cells of the map, $\chi(z_i)$ the winner cell when z_i is presented and $W = \{W_c; c = 1 \cdots k\}$ the set of reference vectors. In this case, the neighboring constraint appears explicitly in the expression of the cost function (1). For two different values of T ($T_1 > T_2$) the "batch" algorithm minimizes two different functions and gives two different local minima. A possible learning

strategy for the "batch" algorithm could be to apply it sequentially by initializing the weights of E_{T_2} by those of the local minimum provided by the minimization of E_{T_1}. If the process is iterated using many minimization steps corresponding to decreasing values of T, each value of T minimizing its related cost function and using as first guess the local minimum reached during the former step, the learning process is similar to the SOM algorithm and can also be considered as a deterministic annealing.
If the learning process is iterated until T* is reached, the cost function $E_{T^*}(\chi, W)$ is thus the k-means distortion function: $E_{T^*}(\chi, W) = \sum_z \sum_c \sum_{C} \| z - W_c \|^2$. The successive minimizations allow to reach a local minimum of E_{T^*} which takes into account the topological constraint.
The similarity between SOM and k-means is rather important since the k-means procedure has a probabilistic interpretation. Under the hypothesis that all the gaussians have the same constant standard deviation k-means is a maximum likelihood algorithm for a mixture density of spherical gaussian distributions [Duda 73].
In the following, we generalize this probabilistic interpretation and propose a probabilistic formalism for topological maps in which each cell c is associated to a gaussian distribution function with mean W_c and covariance matrix $\sum_c = \sigma_c^2 I$. In order to model the neighboring constraint, for a cell c, we define a local mixture which involves the gaussian functions associated to the neighboring cells of c in the map.
We propose a new learning algorithm PRSOM (Probabilistic SOM) which maximizes the likelihood function to estimate the probability density function of the data, under the hypothesis that this probability distribution function is a mixture of the previous local distributions. The PRSOM is valid for any fixed value of T which can be different from T* and avoid the neighborhood to be restricted to one unique cell. In the likelihood function, the parameter T acts as a regularisation term which can have an important role when dealing with applications. Moreover, under some particular hypothesis, the PRSOM algorithm is closely related to the classical SOM and gives a probabilistic interpretation of SOM.
The first section of this paper introduces PRSOM and the relation to SOM. The next section is devoted to simulation and real data results.

2. Probabilistic Self Organizing Map algorithm

Let us first introduce the notations we used. Let D be the data space (D ^{n}) and $A=\{z_i ; i=1,...,N\}$ the training set (A D). We denoted by (C) a map of M neurons, this map is assumed to have a neighborhood system. The distance $\delta(c,r)$ between two neurons (c) and (r) of the map is the length of the shortest path between c and r on the map. The neighborhood size is controlled by a function $K_T(\delta(c,r)) = [1/T]K(\delta(c,r)/T)$ where $K(.)$ is a kernel function. $K_T(.)$ varies with respect to the parameter T which controls the neighborhood size. We associate to each neuron c a gaussian density function f_c with mean the reference vector $W_c = (W_c^1, W_c^2, \cdots, W_c^n)$ and covariance matrix $\sigma_c^2 I$.

2.1 Probabilistic formalism

In the formalism proposed by Luttrel [Luttrel 94], the whole network will be designed as a three layers architecture: the input layer has n neurons receiving the input vector z. The classical map C is duplicated in two similar maps C_1 and C_2 provided with the same topology as C. C_1 and C_2 will be respectively the second and the third layer. The information proceeds from the input layer to C_2.
For a given pattern z, each cell of each layer computes its probable state assuming the Markov property [Luttrel 94]: $p(c_2 / z, c_1) = p(c_2 / c_1)$ and $p(z / c_1, c_2) = p(z / c_1)$. Thereafter, we derive the density function *p(z)* as a mixture of densities defined on C_2:

$$p(z) = \sum_{c_2} p(c_2)\, p_{c_2}(z) \text{ where } p_{c_2}(z) = p(z / c_2) = \sum_{c_1} p(c_1 / c_2) p(z / c_1)$$

The probability density $p(z / c_2) = p_{c_2}(z)$ is a mixture of densities completely defined from the map given the conditional probability $p(c_1 / c_2)$ on the map and the conditional probability $p(z / c_1)$ on the data.. In the following we deal with gaussian densities and assume that:

$$p(c_1 / c_2) = [1/T_{c_2}]\, K_T(\delta(c_1, c_2)) \text{ where } T_{c_2} = \sum_r K_T(\delta(c_2, r)),$$

$$p(z / c_1) = f_{c_1}(z, W_{c_1}, \Sigma_{c_1}),$$

where f_{c_1} is the gaussian density with mean vector W_{c_1} and covariance matrix $\Sigma_{c_1} = \sigma_{c_1}^2 I$.

Under these assumptions $p_{c_2}(z) = [1/T_{c_2}]\sum_{r\ C_1} K(\delta(c_2,r))f_r(z,W_r,\sigma_r)$ is a mixture of gaussians, and the global density $p(z)$ is a mixture of the density $p_{c_2}(z)$ whose parameters have to be estimated. We propose in the following a learning procedure which maximizes the likelihood function, and estimates the mixture parameters.

2.2 Parameters estimation

To deal with the maximization of the likelihood function, we use the Maximum A Posterior (MAP) method and we develop an iterative batch algorithm. The MAP method assumes that each observation z_i is generated by a well defined local mixture p_{c_2}, let χ be the assignment function which assigns each z_i to its local mixture $p_{\chi(z_i)}$. If we assume that the observations are independent, thus the likelihood is:

$$p(z_1, z_2, \cdots, z_N, W, \sigma, \chi) = \prod_{i=1}^{N} p_{\chi(z_i)}(z_i)$$

and the log-likelihood function $E(\chi, W, \sigma)$:

$$E(\chi, W, \sigma) = \sum_{i=1}^{N} -Ln\left(\sum_{r\ C} K(\delta(\chi(z_i),r))f_r(z_i,W_r,\sigma_r)\right) = \sum_{i=1}^{N} E_i(\chi, W, \sigma)$$

according to the assignment functions χ and to the parameters $W = \{W_c; c = 1\cdots k\}$ and $\sigma = \{\sigma_c; c \quad C\}$, the MAP minimizes $E(\chi, W, \sigma)$ in two stages: .*Minimization stage and Assignment stage*. At the k[th] iteration, χ, W and σ will be noted χ^k, W^k and σ^k.

Minimization stage:
During this stage the assignment function χ^{k-1} is kept fixed and the log-likelihood $E(W, \sigma, \chi^{k-1})$ is minimized with respect to W and σ. This is done by using a classical optimization scheme [Duda 73]. The scheme consists in using the parameters W_r^{k-1} and σ_r^{k-1} to estimate the current parameters and setting the derivatives of $E(W, \sigma, \chi^{k-1})$ to zero which yields the following new parameters:

$$W_r^k = \frac{\sum_{i=1}^{N} z_i K(\delta(r, \chi^{k-1}(z_i)) \frac{f_r(z_i, W_r^{k-1}, \sigma_r^{k-1})}{p_{\chi^{k-1}(z_i)}(z_i)}}{\sum_{i=1}^{N} K(\delta(r, \chi^{k-1}(z_i)) \frac{f_r(z_i, W_r^{k-1}, \sigma_r^{k-1})}{p_{\chi^{k-1}(z_i)}(z_i)}}, \quad (\sigma_r^k)^2 = \frac{\sum_{i=1}^{N} \|W_r^{k-1} - z_i\|^2 K(\delta(r, \chi^{k-1}(z_i)) \frac{f_r(z_i, W_r^{k-1}, \sigma_r^{k-1})}{p_{\chi^{k-1}(z_i)}(z_i)}}{n\sum_{i=1}^{N} K(\delta(r, \chi^{k-1}(z_i)) \frac{f_r(z_i, W_r^{k-1}, \sigma_r^{k-1})}{p_{\chi^{k-1}(z_i)}(z_i)}} \quad (2)$$

Assignment stage:
During this stage the parameters W^k and σ^k are kept fixed and a new assignment function is determined in order to improve the log-likelihood $E(W^k, \sigma^k, \chi)$. This new function assigns each observation z_i using the following decision function which defines a new partition of the input space:

$$\chi^k(z) = \underset{c}{\text{argmax}}\ p_c(z) \quad (3)$$

PRSOM algorithm

Initialization: Choose randomly the initial parameters W^0, σ^0 and χ^0.
Minimization step: for a fixed function χ^{k-1} compute the new parameters W^k and σ^k according to (2).
Assignment step: for a fixed parameters W^k and σ^k compute the new assignment function χ^k associated to W^k and σ^k according to (3).
Repeat the iteration step until stabilization.

For a fixed value of T, it can be proved that the PRSOM algorithm converges in a finite number of iterations to a local minimum [Anouar 97].
If we introduce the posterior probability $h_{ic}(r)$ that z_i is generated by the gaussian r, given the mixture $p_c(z_i)$, $h_{ic}(r) = \frac{K(\delta(r,c) f_r(z_i)}{p_c(z_i)}$, at the iteration k formula (2) can be rewritten and gives:

$$W_r^k = \frac{\sum_c \sum_{z_i / \chi(z_i)=c} z_i h_{ic}^{k-1}(r)}{\sum_c \sum_{z_i / \chi(z_i)=c} h_{ic}^{k-1}(r)} \qquad (\sigma_r^k)^2 = W_r^k = \frac{\sum_c \sum_{z_i / \chi(z_i)=c} \|W_r^{k-1} - z_i\|^2 h_{ic}^{k-1}(r)}{\sum_c \sum_{z_i / \chi(z_i)=c} h_{ic}^{k-1}(r)}$$

In these sums, the observation z_i closer to r have a strong weight $K(\delta(r,c) \approx 1$ and the more distant have a weak weight $K(\delta(r,c) \approx 0$ which means that they are not involved in the estimation of W and σ.

Hence, the larger T the more are observations involved in the parameter estimations: we will show in the following experiments that T acts as a regularisation parameter.

2.3 Relation to SOM algorithm

The minimization stage of each iteration of PRSOM can be replaced by a gradient procedure. The use of a stochastic gradient gives for a particular pattern z_i the following "on line " updating rule:

$$W_r^k = W_r^{k-1} - \varepsilon \frac{\partial E_i}{\partial W_r} \text{ and } \sigma_r^k = \sigma_r^{k-1} - \varepsilon \frac{\partial E_i}{\partial \sigma_r} \qquad (4)$$

To enlighten the relation to SOM, we assume that all covariance matrix are equal to σ^2 I where σ is a constant and that the kernel function is constant on a predefined neighborhood: $1/N_c$ if $\delta(r,c) \le d$ and zero otherwise, where N_c is the number of cells r such that $\delta(r,c) \le d$. The updating rule (4) becomes:

$$\begin{cases} W_r^k = W_r^{k-1} - \varepsilon \dfrac{1}{\sigma^2} \left(\dfrac{f_r(z_i)}{\sum_{c/\delta(r,\chi^{k-1}(z_i)) \le d} f_c(z_i)} \right) (W_r^{k-1} - z_i) & \text{if} \quad \delta(r,\chi^{k-1}(z_i)) \le d \\ W_r^k = W_r^{k-1} & \text{otherwise} \end{cases} \qquad (5)$$

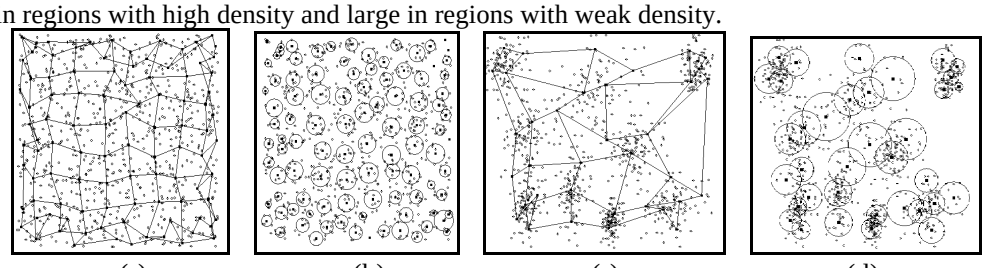
In this case the "on line " PRSOM is similar to the SOM updating rule. The main difference is that the gradient rate is determined with respect to the kernel function in the data space and not with respect to the distance on the map space. Furthermore, if we assume that the constant σ is large enough, the "on line " PRSOM and SOM become similar: SOM is a PRSOM in which all covariance matrix are the same and equal to σ^2 I where σ is sufficiently large.

3. Simulation results

We first present some simulations which show that PRSOM preserve the topological order and gives topological maps which are similar to those provided by SOM. We will discuss afterwards, the use of a supervised version of PRSOM to solve regression task.

3.1 Topology preserving

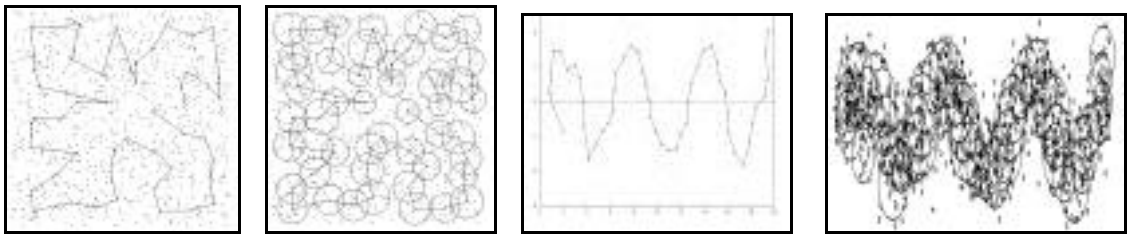
In the first experiment, we simulate two different data sets: the first one has 800 examples generated according to the uniform density in D=[-12,12][-12,12]; the second data set has 900 examples and is generated according to a mixture of 9 gaussians. For all the simulations we use a map with square neighborhood. The kernel function is defined by: $K_T(u) = [1/T] e^{[-u^2/T^2]}$ where the parameter T represents the width of the Kernel. For the first data set the map has 10x10 neurons, and for the second it has 6x6 neurons. For each experiment, we display reference vector and the resulting maps after learning.
Figures 1(a) and 1(c) show the maps for the two data sets after training. We can observe the topological order of the map. Figures 1(b) and 1(d) show the estimated standard deviation represented by the radius of a circle centered on the reference vector. One circle represents the "influence region" of the associated reference vector. One can see the influence of the σ parameters, they fit into the local distribution: they are small in regions with high density and large in regions with weak density.



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

Figure 1. (a) Represents the final map after learning with PRSOM algorithm for the first data base (uniform distribution). (b) Displays standard deviation as a circle centered on the reference vector. (c) Represents the final map after learning with PRSOM algorithm for the second data base (9 Gaussians). (d) Displays standard deviation as a circle centered on the reference vector.

In the second experiment, we use a one dimensional topological map with 50 reference vectors and apply PRSOM on the first data set (uniform data). Figure 2(a) represents the map and the data. figure 2(b) represents the standard deviations centered on the reference vector.
We generate now a third data set which lay on 1-D variety. We generate 1000 noisy data such that: $z_i = \sin(x_i) + \varepsilon_i$, ε_i are generated according to the normal density N(0; 0.1). Figure 2(c) shows that the map approximate well the sinus function. Figure 2(d) represents the standard deviations.



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

Figure 2. One dimensional map using PRSOM with T=1

The intrinsic dimension of the data is retrieved by the estimated standard deviations. The four examples show that the standard deviations allow to cover all the data and thus to retrieve the real data dimension.

3.2 Experiments on oceanographic data

In order to draw an accurate map of sea surface temperature, oceanographers need to establish spatial regression for data collected since the beginning of the century. In this application we deal with sea surface temperatures collected in July on the European coasts. Data are supplied by The Marine Hydrographic and Oceanographic Service (SHOM). Each observation consists of a measure of the sea surface temperature and its location (longitude and latitude). The learning set which involves 4200 samples is represented in figure 3(a). The validation set concerns 3220 samples. We run PRSOM using the learning set on a squared grid map of 20x20 reference vectors represented by the figure 3(b).

(a) (b)

Figure 3: (a) Dots represent the observation in longitude and latitude of the learning set
(b) Represents the final map after learning with PRSOM (T=1)

To achieve regression task, we use a supervised version PRSOM-RBF of PRSOM based on RBF (radial basis function) network [Moody 89] [Anouar 97] in which the basis function are given by PRSOM. We select the best parameter T using a cross validation on the validation set. Table 1 gives the rms (root mean square) error for different values of T. The smoothing effect of T appears clearly on the figure 4. We run on the same data the algorithm NGBF (Normalized Gaussian Basis Function) proposed by Nowlan [Nowlan 90] optimizing the number of reference vectors. The best result of NGBF has been obtained for 20 reference vectors.

	PRSOM-RBF:T=0.5	PRSOM-RBF:T=1	PRSOM-RBF:T=2	NGBF
rms on validation set	1.19	1.13°	1.22°	1.22°

Table1: Performances of PRSOM-RBF and NGBF on the sea surface temperature

(a) (b)

Figure 4: (a) Isobars of the sea surface temperature given by the topological map figure 3(b) for T=1

(b) Isobars of the sea surface temperature for T=2

Figure 4(a) provides realistic isobars which are compatible with those given by techniques used by oceanographers. We can see that the parameter T governs the level of smoothness of the isobars.

4. Conclusion

We have developed a new probabilistic learning algorithm for topological map which estimates parameters of a mixture of densities using likelihood estimation. Thus unlike the SOM algorithm, PRSOM optimizes an objective function defined by the likelihood function. PRSOM depends on a regularisation parameter T which has to optimized. We show that this algorithm allows a probabilistic interpretation of topological maps. Then we investigated the relationship between an on-line version of this algorithm and the classic SOM algorithm. We presented an application of the PRSOM-RBF algorithm to regression task with the aim to reconstruct the sea surface temperature function. The effect of T appears clearly: the parameter T governs the smoothness of the output function.

Acknowledgments

This work was partially supported by the EPSHOM 87470 contract.

References

[1] Anouar F., Badran F., Thiria S, 1997, "Probabilistic self Organizing Map: Application to Classification" ESANN, to appear.
[2] Duda R., Hart P., 1973. ″Pattern Classification and Scene Analysis″, New York: Wiley.
[3] Luttrel S.P , 1994. "A Bayesian Analysis of Self-Organizing Maps". Neural Computing vol 6,
[4] Martinez T., Berkovich S., Shulten K., 1993. "Neural-gas, network for vector quantization and its application to time series prediction", IEEE trans. Neural Networks 4, 558-569.
[5] Moody J., Darken C., ″Fast learning in networks of locally-tuned processing units″, Neural Comp. 1, 1989.
[6] Nowlan S., ″Maximum likelihood competitive learning″, Proceeding of Neural Information Processing Systems, 574-582, 1990.
[7] Kohonen T. 1994. Self-Organizing Maps. Springer.