Helsinki University of Technology
Publications in Computer and Information Science

Report E8
January 2007

DESCRIPTION OF INPUT PATTERNS BY LINEAR MIXTURES OF SOM MODELS

Teuvo Kohonen

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY
TECHNISCHE UNIVERSITÄT HELSINKI
UNIVERSITE DE TECHNOLOGIE D'HELSINKI

Distribution:
Helsinki University of Technology
Department of Computer Science and Engineering
Laboratory of Computer and Information Science
P.O. Box 5400
FI-02015 TKK, Finland
Tel. +358-9-451 3267
Fax +358-9-451 3277

This report is downloadable at
http://www.cis.hut.fi/Publications/

ISBN 978-951-22-8614-0
ISSN 1796-2803

Description of Input Patterns by Linear Mixtures of SOM Models

Teuvo Kohonen

Helsinki University of Technology
Adaptive Informatics Research Centre
Espoo, Finland
P.O. Box 5400, FIN-02015 HUT

Abstract. This paper introduces a novel way of analyzing input patterns presented to the Self-Organizing Map (SOM). Instead of identifying only the "winner," i.e., the model that matches best with the input, we look for the linear mixture of the model (reference) vectors of the SOM that approximates to the input vector best. It will be shown that if only nonnegative weights are allowed in this linear mixture, the expansion of the input pattern in terms of the models is very meaningful and contains only the essential terms. Especially if the input pattern contains information about a mixture of objects, as the case is when the input describes a mixed state in a process or a machine, or if the input consists of features from several independent objects, this fitting gives more exact information about, and provides a better insight into the input state than what the mere "winner" can give.

Keywords: least-squares fitting, linear mixture, self-organizing map

1. Introduction

When an unknown input vector is presented to the *Self-Organizing Map*, called briefly the *SOM* (Kohonen 2001), the algorithm returns and displays the location of the best-matching model, the "winner." In the case that the input vectors form a sequence, the corresponding best-matching models can be thought to constitute a trajectory, which describes and visualizes, e.g., the temporal development of the state of a process or a machine. Additional information about the degree of matching is contained in the quantization errors, i.e., the distances of the input vectors from the model vectors that are closest in the input space.

However, it has long been felt that a single response, the "winner," to an input pattern is not sufficient for the description of input information that is derived from two or more independent sources, for instance when the input describes features of several simultaneously occurring distinct objects. One might stipulate that separate responses, corresponding to the different objects, should be obtainable. Consider also that if the

SOM is used to track sequences of the states of a system or a machine, the input vectors are usually constructed as averages of measurements made over time windows of considerable length, say, minutes or even hours. If the system is just then undergoing a state transition, the input vector averaged over the time window is expected to be a linear mixture of the samples, eventually representing different states that have been occurred within that window.

Another reason for mixtures of input states to occur is that the system from which the input observations are derived is defined by several more or less independent state variables. Their deviations from regular values are caused by faults or noise. The simultaneous occurrence of two or more independent faults in different parts of the system might then be reflected as a mixed input state of the SOM, and thus as a mixture of input signal components, each of which corresponds to an individual fault.

A very intriguing application occurs in document analysis that is based on the usage of typical words in the text. An interdisciplinary or a multidisciplinary document is expected to contain words from the different vocabularies of its subtopics. One may then be interested in the relative contributions of the mixed topics in the document.

Consider also that if the input vector is, say, close to two model vectors in the input space, it is a haphazard choice which one of the latter will be selected for the "winner." Then, too, it would be desirable to evaluate the probabilities of the best candidates for the "winner."

It will be shown in this article that a more thorough description of the input is obtainable if, instead of determining only the "winner" among the models, one is able to fit a *linear mixture of the models* (reference vectors) to the input. It should be noted that I do not mean "K winners" that are rank-ordered according to their matching, nor a set of "parallel winners," each of which is defined over a local area of the SOM. Instead, the input pattern is approximated by the linear mixture of *any* subset of models (i.e., reference vectors) that approximates to the particular input best.

In this paper we shall show, however, that if only *nonnegative weights* are allowed in the fitting of the models, there will be left only a relatively very small number of models in the mixture, i.e., the fitting is very selective.

Since we are going to deal with linear-fitting problems in the sequel, it may seem proper that the SOM should be of the *dot-product* type, in which the matching of the input vector with the model vectors is measured in terms of their dot products. Then the input vector and all of the weight vectors shall be *normalized*, say, to unit length. Nonetheless there are no restrictions in principle to the application of the same ideas to SOMs that have been constructed on the basis of the Euclidean or any other metric. Even in these cases, however, before using this method, it will be mandatory to normalize the SOM model vectors.

Let us regard the input as a Euclidean vector **x**, and let its dimensionality be *n*. In matrix calculations **x** shall be regarded as an *n*-dimensional *column vector*. Let us then denote the SOM by a *p* times *n* matrix **M**, where *p* is the number of the *models*. If the *model vectors* (regarded as *column vectors*) of dimensionality *n* are denoted as $\mathbf{m}_i$ = 1, 2, … , *p*, they constitute the *rows* of **M** and must then be denoted by $\mathbf{m}_i'$**,** where the prime (´) denotes the *transpose* of a vector or a matrix. All of the $\mathbf{m}_i$ shall have an identical Euclidean norm. In the *dot-product SOM*, the "activation" of the models, or the *degree of matching* of **x** with the $\mathbf{m}_i$, is though to be represented by the *p*-dimensional *activation vector* (column vector) **y**,

$$\mathbf{y} = \mathbf{M}\,\mathbf{x} \,. \tag{1}$$

The largest component of **y** identifies the best-matching model, the "winner."

Comment. Graphically, the SOM is most often defined as a two-dimensional array of nodes where with each node *i*, a model $\mathbf{m}_i$ is associated. During learning, the models in this array interact in such a way that the highest-activated cell imposes corrections on its neighboring models in the array in the same direction. One must clearly realize the distinction between the *rows* of *matrix* **M** and the *2-D geometry* of the *SOM array*, however.

2. Failure of the unconstrained linear fitting

Let us try to fit the *best linear mixture* of any given *reference vectors* $\mathbf{m}_i$ to a given vector **x**. In other words, we want to determine the optimal scalar coefficients k_i in the following error expression **e**, whereupon the Euclidean norm of **e** shall be minimized:

$$\mathbf{e} = k_1\,\mathbf{m}_1 + k_2\,\mathbf{m}_2 + \ldots + k_p\,\mathbf{m}_p - \mathbf{x} \,. \tag{2}$$

Let **k** be the *column vector* formed of the k_i,

$$\mathbf{k} = [k_1, k_2, \ldots, k_p]' . \tag{3}$$

The linear mixture of the $\mathbf{m}_i$ can be written, using matrix expressions, as

$$k_1\,\mathbf{m}_1 + k_2\,\mathbf{m}_2 + ... + k_p\,\mathbf{m}_p = \mathbf{M}'\,\mathbf{k} \,, \tag{4}$$

where $\mathbf{M}'$ is the transpose of the matrix **M** that has the reference vectors as its rows; the latter are identified as the $\mathbf{m}_i'$. Now $\mathbf{M}'\,\mathbf{k}$ is the *estimate* of **x**. If the fitting error is written as

$$\mathbf{e} = \mathbf{M}'\,\mathbf{k} - \mathbf{x} \,,$$

the square of the Euclidean norm of **e** is

$$\mathbf{e}' \mathbf{e} = \mathbf{k}' \mathbf{M} \mathbf{M}' \mathbf{k} - \mathbf{k}' \mathbf{M} \mathbf{x} - \mathbf{x}' \mathbf{M}' \mathbf{k} + \mathbf{x}' \mathbf{x} . \quad (5)$$

It is generally known that $\mathbf{e}'\mathbf{e}$ is minimized if its *gradient* with respect to **k** is equal to zero. This gradient (cf., e.g., Kohonen, 2001, Ch.1) is

$$\mathrm{grad}_{\mathbf{k}} (\mathbf{e}' \mathbf{e}) = \mathbf{M} \mathbf{M}' \mathbf{k} - \mathbf{M} \mathbf{x} . \quad (6)$$

From the expression (6) we may try to solve for **k**:

$$\mathbf{k} = (\mathbf{M} \mathbf{M}')^{-1} \mathbf{M} \mathbf{x} . \quad (7)$$

Unfortunately, the expression (7) can only be computed if $(\mathbf{M} \mathbf{M}')^{-1}$ exists, i.e., if the determinant of $\mathbf{M} \mathbf{M}'$ is nonzero. A necessary condition for it is that all of the $\mathbf{m}_i$, $i = 1, 2, \ldots, p$ are *linearly independent*. For the SOM matrices this is usually not the case.

Even though we would have a SOM in which the dimensionality of the vectors and the number of the models were identical, and even though $(\mathbf{M} \mathbf{M}')^{-1}$ would exist, we may discern a weird result. If the input **x** is an arbitrary vector, some of the k_i in its linear mixture may attain very large values (say, some thousands when the vectors are normalized). *The fitting may be perfect, but it makes no sense.*

This is a kind of a problem that is called "overfitting." Such an oddity can be explained by the following simple example. Consider that usually the neighboring SOM model vectors are rather similar, i.e., their inner products are small. In Fig. 1 we have two model vectors $\mathbf{m}_i$ and $\mathbf{m}_j$ that have a small mutual angle. Their weighted sum has to approximate to a vector **x** that is very roughly perpendicular to them. In this example, one of the fitting coefficients (k_i) attains a very large and positive value, whereas the other one (k_j) has a very large negative value.

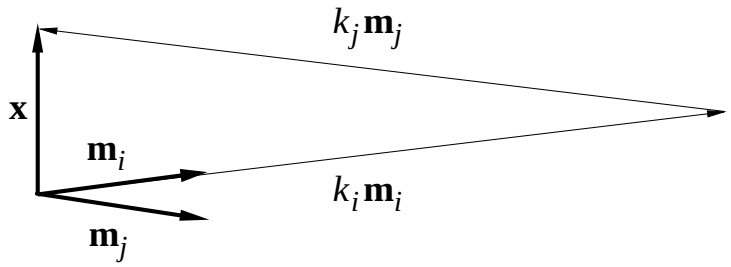

Fig. 1. Explanation of the "overfitting" effect.

3. Fitting with nonnegative weighting coefficients

Much attention has recently been paid to least-squares problems where the fitting coefficients are constrained to *nonnegative* values. Such constraints are natural, when the negatives of the samples have no meaning, for instance, when the input consists of statistical indicators that can have only nonnegative values, or is a weighted word histogram of a document. In these cases at least, the constraints contain additional information that is expected to make the fits more meaningful. In any case we are able to circumvent the above "overfitting" problem, if we forbid the (eventually large) negative weights.

The mathematical problem is formulated as follows: for general dimensionalities of **M** and **k**,

$$\text{minimize } \text{norm}(\mathbf{M}'\,\mathbf{k} - \mathbf{x}) \tag{8}$$

subject to the condition that all of the elements of **k** are nonnegative. In this work the norm is Euclidean.

Gradient-descent optimization. There exist several ways for the solution of (8). The simplest and most straightforward is the *gradient-descent optimization*. An iterative algorithm that takes into account the nonnegativity constraint can be specified in the following simple way. Denoting the component i of the gradient in eq.(6) by G_i we write

$$\text{for all } i\,, \quad k_i\,(t+1) = \max\,(0,\, k_i\,(t) - \mathit{alpha}\; G_i\,)\,, \tag{9}$$

where t is the integer-valued index of the iteration step, and *alpha* is a scalar factor that defines the size of the gradient step. Typically, a few tens of thousands of iteration steps (9) are necessary to reach a reasonably stable solution with a numerical accuracy of, say, three significant digits.

It may still be possible to optimize (9).

The Matlab function *lsqnonneg*. The present fitting problem belongs to the broader category of *quadratic programming* or *quadratic optimization*, for which numerous methods have been developed over the years. A recent one-pass solution of (8) is based on the *Kuhn-Tucker theorem* (cf. Lawson & Hanson, 1974), but it is too complicated to be reviewed here. Let it be mentioned that it has been implemented as the function named the *lsqnonneg* in the Matlab:

$$\mathrm{K} = \mathrm{lsqnonneg}\,(\mathrm{M}', \mathrm{X}, \mathrm{K}(1))\,, \tag{10}$$

where **M** has been denoted by M and **x** by X, respectively, and K**(**1**)** is the initial value of **k** used by the algorithm.

Initialization. It might seem advisable, in order to guarantee a fast convergence to the global optimum, to start with a good initial value of the coefficient vector. In the SOM, such an initial "linear mixture" could be taken to consist of the "winner" term $\mathbf{m}_c$ only. In other words, $k_c = 1$, whereas the rest of the k_i are put equal to zero. In the gradient-descent method this initialization is slightly better than when starting with the zero vector for $\mathbf{k}(1)$. Contrary to that, with the *lsqnonneg*, no significant difference in the convergence times with different initializations has been observed.

Limitation of the methods. For both of the above methods, the *rank* of the matrix $\mathbf{M}$ can be arbitrary, and solutions exist even though $(\mathbf{M}\mathbf{M}')^{-1}$ does not. Nonetheless one can easily see that there are theoretical cases where no solutions exist, or where the optimal solution is not unique. The latter case is due when some of the $\mathbf{m}_i$ *in the final optimal mixture* are linearly dependent. It can be checked by first forming the matrix $\mathbf{M}_1$ of those models, the corresponding coefficients k_i of which are positive, and then computing the determinant $\det(\mathbf{M}_1 \mathbf{M}_1')$
If it is zero, the components in the linear mixture are linearly dependent. Such a case, and already the case that the components are nearly dependent, will also be manifested by a slow-down of the computing time in both of the above methods

Comparison of the computing times of gradient descent and *lsqnonneg*. In the above methods, the computing time depends strongly on $\mathbf{x}$ and $\mathbf{M}$. Also the use of multiple cache memory systems in the contemporary computers makes the benchmarking of computing time very vague.

The following experiments were made with $\mathbf{x}$ and $\mathbf{M}$ consisting of positive random numbers. The matrix $\mathbf{M}$ was square (i.e., the number of models and their dimensionality were equal), and the dimensionality ranged from 10 to 1000. The results given in Fig.2 are medians of five independent runs. The number of iterations in gradient descent was 10 000.

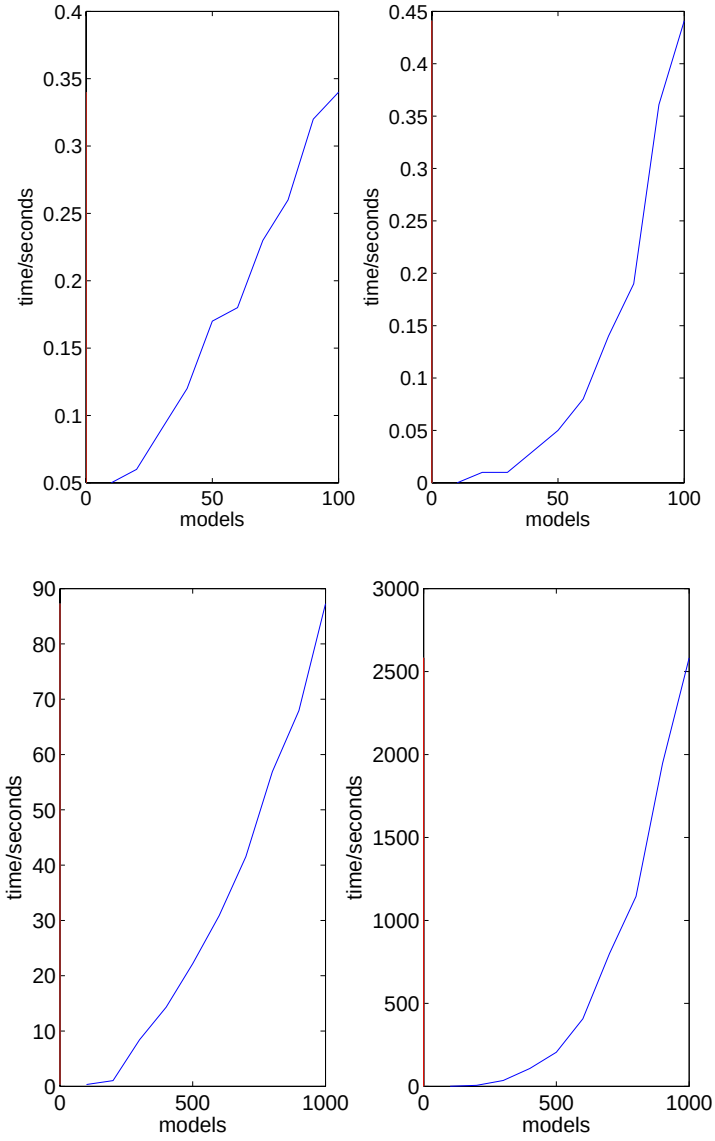

Fig. 2. Comparison of computing times. Left: the gradient-descent method. Right: the function *lsqnonneg*.

4. Applications

4.1. Cellular-Phone Data

The first practical example describes the performance of a cell in a cellular-telephone network. The input vector to the SOM was defined by 22 variables that describe the *key performance indices (KPI*) such as signal qualities in inward and outward transmission, frequencies of breaks in operation relating to different kinds of faults, and loadings of the

cell. We had data from 110 cells available, and each one of the records was an average of the respective measurement or evaluation over an hour. This example is from cell No. 50 during 879 hours of uninterrupted operation.

The particular SOM constructed for this study consisted of 80 models with the dimensionality of 22.

A comparison of the different responses to the input vector has been presented in Fig. 3 for five successive sampling intervals (Nos. 10 through 14). The algorithm thereby used was the *lsqnonneg*. On the first row we see the degree of matching (dot product) of the various models with the input. The second row shows the location of the ”winner” on the SOM. The third row illustrates the weighting coefficients k_i, displayed on the SOM groundwork. The saturation of the color corresponds to the value of k_i. One can discern that the resulting mixtures that describe the input states are not very complex: on the average, they contain only about five per cent of all models.

Table 1 shows the quantization and fitting errors for five samples that were not involved in the computation of the SOM. Let us recall that the expression of the former error is norm($\mathbf{m}_c - \mathbf{x}$), where $\mathbf{m}_c$ is the “winner,” and that of the latter is norm($\mathbf{M}\mathbf{k} - \mathbf{x}$), respectively.

Table 1. The quantization error and the fitting error for five successive sampling intervals of the mobile-phone data.

Interval No.	Quantization error	Fitting error
10	.5659	.3072
11	.2996	.2272
12	.1777	.1270
13	.1550	.0907
14	.1694	.1435

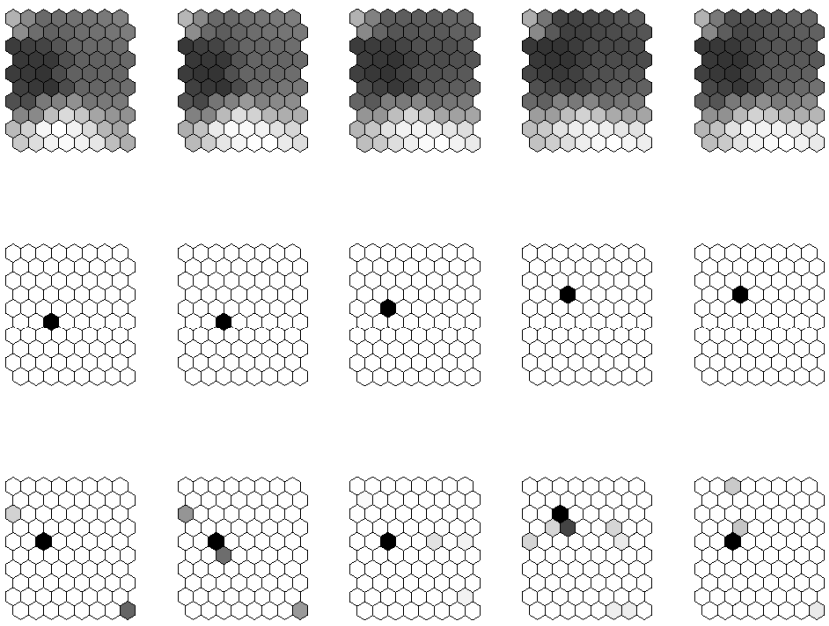

Fig. 3. Comparison of different displays. Horizontal direction: five successive sampling intervals. First row: The activation vectors **y** = **Mx** averaged over one-hour sampling periods. Second row: The locations of the "winner" on the SOM. Third row: the fitting coefficients k_i, shown on the SOM groundwork.

4.2. Document Analysis

In the second application, the input item was a weighted histogram of the vocabulary of a given document. The objective was to perform a *text analysis*, in order to discover to what extent the vocabularies of the given document and those of the other documents overlap. It must be emphasized that only the *most distinctive words* were taken into account in the overlapping vocabularies.

The Reuters corpus. The first more extensive experiment of this type was based on the text corpus collected by Reuters. No original documents were made available, but Lewis et al. (2004) have preprocessed the textual data by removing the *stop words*, and reducing the words into their *stems*. J. Salojärvi from our laboratory selected a 4000-document subset from this preprocessed corpus, restricting only to such documents that were classified into one of the following categories:

1. Corporate-Industrial.
2. Economics and Economic Indicators.
3. Government and Social.
4. Securities and Commodities Trading and Markets.

Salojärvi then picked up those 1960 words that appeared at least 200 times in the selected data and weighted the word *i* (”term”) of document *j* by the factor

$$w_{ij} = (1 + \log(TF_{ij})) \log(N / DF_i), \quad (12)$$

where TF_{ij} is the ”term frequency” (frequency of word *i* in document *j*), DF_i (”document frequency”) tells in how many documents word *i* appears, and *N* is the total number of documents (Manning & Schütze,1999).

In order to carry out statistically independent experiments, a small subset (one per cent) of the documents was set aside for testing. Using the weighted word histograms of the rest of the 4000 documents as input, a 2000 by 1960 SOM was constructed. Fig. 4 shows the four *hit histograms*, where the samples of the various classes were separately presented to the input of the final SOM.

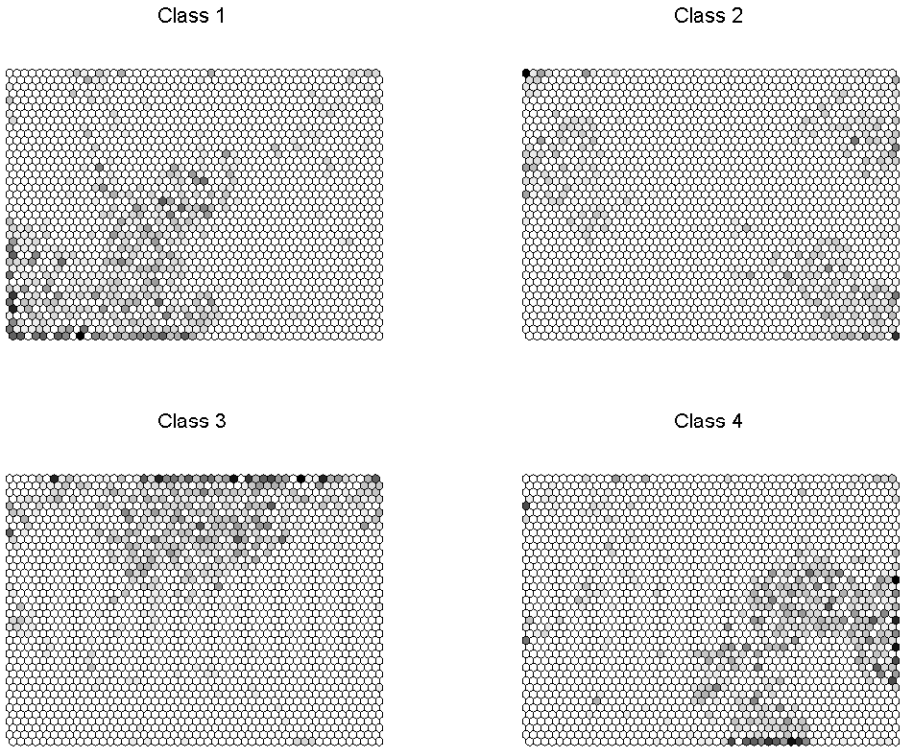

Fig. 4. Distributions of hits of the samples of the four classes on the SOM, respectively. Class 1: Corporate-Industrial. Class 2: Economics and Economic Indicators. Class 3: Government and Social. Class 4: Securities and Commodities Trading and Markets.

Labeling of the SOM models. The number of input items used for training was only about twice the number of models, so it was not reasonable to label the models according

to the majority of hits on them. Instead, each model was labeled according to its *K* nearest neighbors in the input space of the training data, where *K* was initially taken as 10 and increased gradually only in the case of *ties* in the determination of the majority of labels. Fig. 5 shows the labeled class regions on the SOM, using pseudocolors.

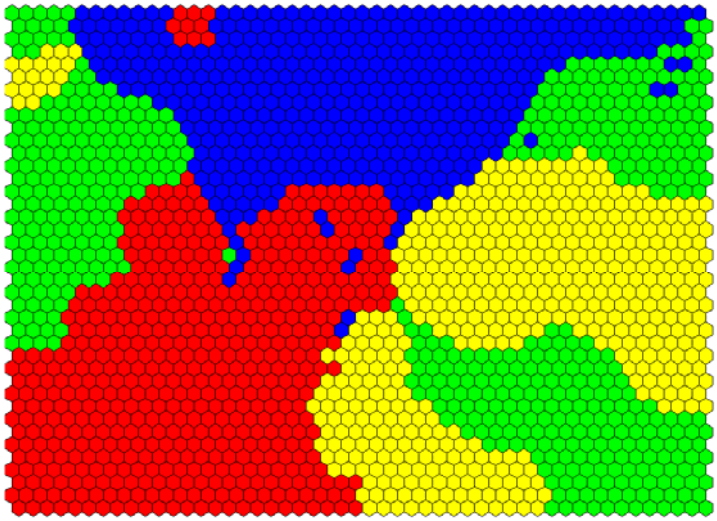

Fig. 5. Labeling of the SOM nodes according to the document classes. Red: Corporate-Industrial. Blue: Economics and Economic Indicators.Green: Government and Social. Yellow: Securities and Commodities Trading and Markets.

Fitting results. The best-fitting linear mixtures of models for four documents are shown in Fig. 6 on the SOM groundwork, indicating the classification and the value of the due k_i by the hue and the saturation of the pseudocolor of the corresponding SOM location, respectively.

The quantization error and the fitting error for the four documents are given in Table 2. Notice that these documents were excluded from the corpus when training the SOM.

Table 2. The quantization error and the fitting error for four documents

Document No.	Quantization error	Fitting error
101	.8541	.8036
201	.8084	.7767
501	.8631	.8303
901	.8686	.8082

Class mixtures. If it is wanted to evaluate, e.g., the degree of *multidisciplinarity* of a document, one approach is to sum up the k_i of each class separately. Fig. 7 illustrates, in relation to this approach, the affiliations of the individual documents with the classes 1 through 4, respectively.

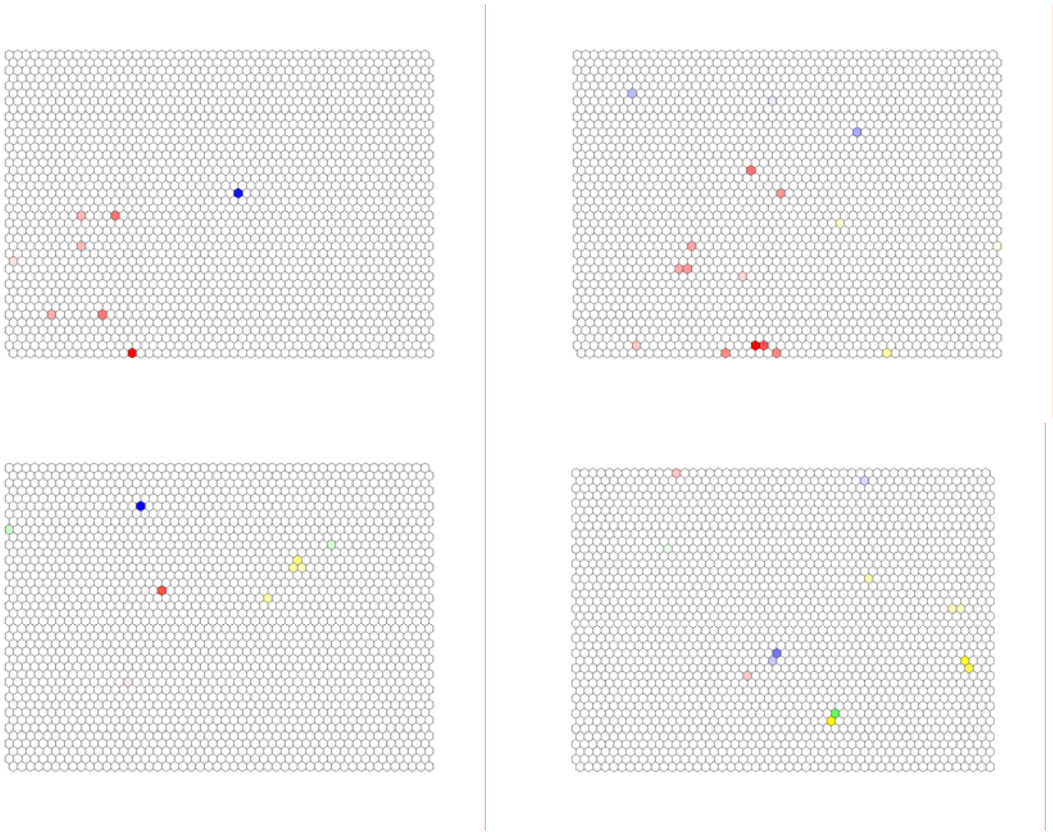

Fig.6. Optimal linear mixtures (with nonnegative coefficients) of models for four documents.

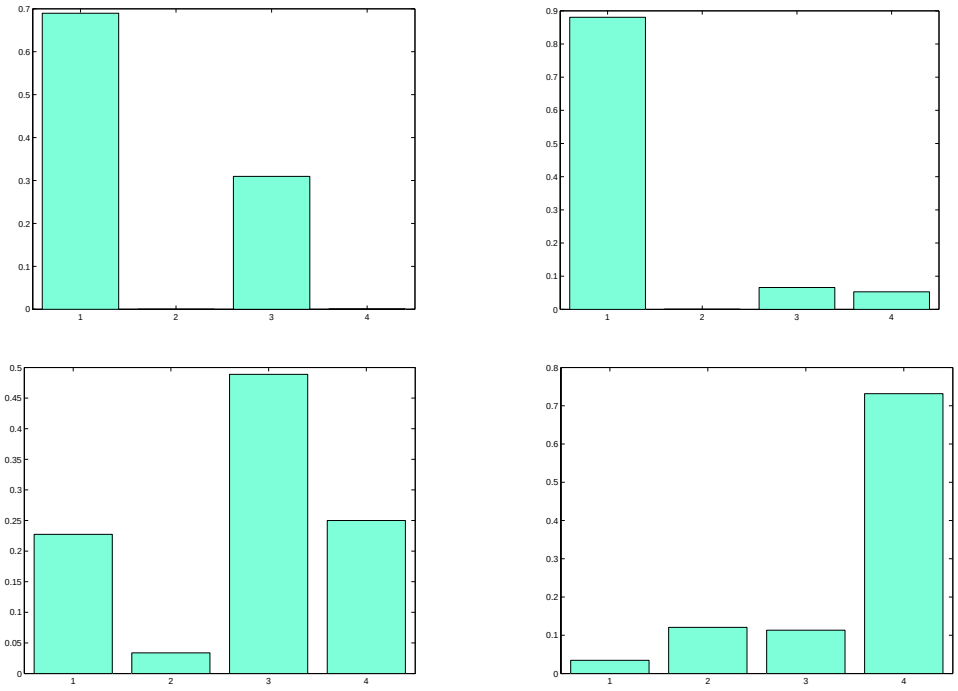

Fig. 7. Affiliations of the four documents with the four given classes. The sum of the bars in each subfigure is normalized to unity.

5. Crucial test

This paper is finally provided with a crucial test to demonstrate the degree of selectivity in finding out the components in the linear mixture. To that end the input vector **x** was formed as a sum of p models, $p < n$, which were picked up at random from all of the models, such that no model occurred twice or more often in this mixture. Then **x** was normalized. The SOM was the same as that used in Subsection 4.2. The fitting function was the *lsqnonneg*.

Table 3 gives the value of the determinant det($\mathbf{M}_1$ $\mathbf{M}_1'$) and the computing time (in seconds) for different numbers of models selected to the mixture that makes up **x**. *The fitting error in all of these cases was zero within the numerical accuracy of computation.*

Table 3. The determinant (see text) and the computing time for different numbers of models in x

Models	Determinant	Computing time (seconds)
5	.0011	3 566
10	3.1×10^{-6}	10 258
50	3.3×10^{-44}	49 346
100	2.4×10^{-105}	100 170

As one can see, the value of the determinant (which represents the degree of linear dependence of the components) is strongly reflected in the computing time. One may make a comparison with the computing times in Subsection 4.2., in which the number of components in the linear mixture was at most 18, and the computing time was at most 3084 seconds, respectively.

Finally, the fidelity of the fitting result (**k**) with **x** consisting of 100 models is demonstrated in Fig. 8.

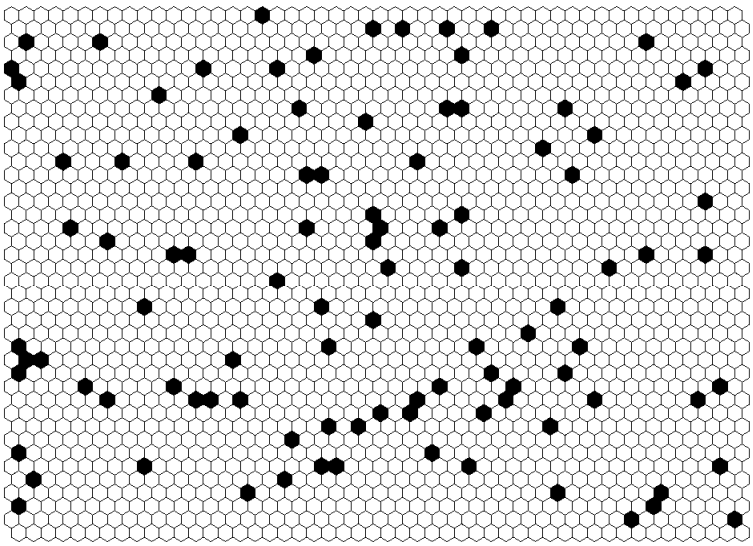

Fig. 8. This picture identifies the 100 models included in **x**, and also displays the vector k (linear mixture) of the models fitted to **x**. The fitting error was zero within the computing accuracy.

6. Conclusion

The purpose of this presentation has been to extend the use of the SOM by showing that instead of a single ”winner,” one can define several “outputs” that together describe the input pattern more accurately. These “outputs” were defined to be the components in the linear mixture of SOM models that approximate to the input best in the sense of least squares. Only nonnegative weights in the fitting were allowed.

In the light of the above experiments it looks evident that the approximation of the input by the optimal nonnegative-coefficient linear mixture of the SOM models contains more information than the mere location of the best-matching model can give. It is striking how few components are then needed.

If the models fall into classes that are known *a priori*, the weights of the models in the mixture can be interpreted as expressing the degree to what the input is affiliated with the various classes.

Acknowledgements.

The author is obliged to the following persons: Kimmo Raivio and Jarkko Salojärvi for making their data available for these experiments, and Lasse Lindqvist for helping in the processing of the data. This work was made under the auspices of Academy of Finland and Helsinki University of Technology.

References

Kohonen, T., *Self-Organizing Maps,* 3rd Edition, Springer-Verlag, 2001.

Lawson, C.L. and Hanson, R.J., *Solving Least-Squares Problems,* Prentice-Hall, 1974.
Lewis, D.D., Yang, Y., Rose, T.G., and Li, F.: RCV1: A New Benchmark Collection for Text Categorization Research. *J. Machine Learning Research* **5**, 361-397, 2004.

Manning, C.D. and Schütze, H., *Foundations of Statistical Natural Language Processing,* MIT Press, Cambridge, Mass., 1999

(Communicated on January 15, 2007; revised February 13, 2007))