Stochastic Self-Organizing Map variants with the R package SOMbrero

Nathalie Villa-Vialaneix

Stochastic Self-Organizing Map variants with the R package **SOMbrero**

Nathalie Villa-Vialaneix MIAT, Université de Toulouse, INRA
BP 52627, 31326 Castanet Tolosan cedex, France
Email: nathalie.villa-vialaneix@inra.fr
Telephone: +33 (0)5 61 28 55 73

(Invited Paper)

Abstract—Self-Organizing Maps (SOM) [1] are a popular clustering and visualization algorithm. Several implementations of the SOM algorithm exist in different mathematical/statistical softwares, the main one being probably the SOM Toolbox [2]. In this presentation, we will introduce an R package, **SOMbrero**, which implements several variants of the stochastic SOM algorithm. The package includes several diagnosis tools and graphics for interpretation of the results and is provided with a complete documentation and examples.

1 Introduction

Self-Organizing Maps (SOM), introduced by Teuvo Kohonen [1], [3], [4], are a popular clustering and visualization algorithm. The success of the method is due to its very simple definition as well as its ability to perform, in a single analysis, clustering and visualization by projection of the data onto a small dimensional grid. This is a powerful and easy-to-use method for the exploration of multidimensional data. Originally, SOM has been inspired by neuro-biological learning paradigms which were intended to model some sensory or cognitive processes in which the learning is directed by the experience and the external inputs without supervision. Later, the method has been used in a wide variety of areas ([2] reports more than 7,000 scientific publications related to SOM with applications in industrial, biomedical and financial analyses, for instance).

Several implementations of the SOM algorithm exist in different mathematical/statistical softwares, most on them usable only for numeric data. The main one is probably the SOM Toolbox[1], which is released under GPL2 licence and runs in Matlab. The toolbox contains many different functions to learn and interpret SOM and to handle various types of data, including datasets with missing data, symbol strings datasets and relational datasets (*i.e.*, datasets described by a dissimilarity, [5]). It also includes implementations of Generative Topographic Mapping [6] and of LVQ and metric learning for LVQ [7], [8]. The toolbox is fully documented and illustrated in the book [2].

Since the free statistical software R [9], is one of the most (if not the most) widely used program in the statistical and bioinformatics communities, we have implemented a full R package providing various versions of the SOM algorithm and many functions for diagnosis and interpretation. **SOMbrero** is inspired by Patrick Letremy's SAS programs[2]. Prior **SOMbrero**[3], several R packages were proposing SOM implementations (*e.g.*, **class**, **som**, **popsom**, **kohonen**, ... see [10] for a more precise description of these packages). However, as far as we can tell, **SOMbrero** is the most complete one, with user-friendly features and implementations of variants of the original algorithm for relational data and contingency tables, based on the stochastic version of the method. Current version of the package is 1.2 (2016/09/02) and the package runs on R ($\geq$ 3.0.1) with dependencies to the packages **wordcloud**, **scatterplot3d**, **RColorBrewer**, **shiny**, **grDevices**, **graphics** and **stats**.

The next sections describe the different features of the package. More precisely Section 2 presents the general organization of the package and its standard use for numeric datasets. Section 3 presents applications of SOM to explore contingency tables and Section 4 describes a variant for dissimilarity datasets.

2 SOM for numeric datasets and general organization of SOMbrero

This section provides an overview of the package main features. It fully describes functions that can be used for the standard SOM (that processes numeric datasets). However, most of what is described in this section is not restricted to numeric datasets and also exist for the other two variants of the algorithm that are described in Sections 3 and 4.

2.1 Training stochastic SOM for numeric datasets

SOMbrero implements the stochastic (also called "on-line") version of the SOM algorithm. More precisely, the dataset

1. current version 2.1, last update from December 2012, available at http://research.ics.aalto.fi/software/somtoolbox/

2. current version 9.1.3, last updated in December 2005, no longer maintained, available at http://samos.univ-paris1.fr/Programmes-bases-sur-l-algorithme

3. https://CRAN.R-project.org/package=SOMbrero

978-1-5090-6638-4/17/$31.00 ©2017 IEEE

$\mathbf{X} = (x_{ij})_{i=1,\dots,n, j=1,\dots,d}$, made of n observations taking values in $\mathbb{R}^d$, is mapped into a low dimensional grid composed of U units. Conversely, every unit u is associated with a prototype $p_u \in \mathbb{R}^d$. The grid induces a natural distance d on the map which provides a measure of dissimilarity between every pair of units, $d(u, u')$. The stochastic version of the algorithm processes the observations one by one, with the iterative application of two steps:

- an *assignment step* where one observation (on-line version) is classified into the unit with the closest prototypes (according to $\mathbb{R}^d$ Euclidean distance);
- a *representation step* where all prototypes are updated according to the new assignment. For the on-line version of the algorithm, this step is performed by mimicking a stochastic gradient descent scheme.

The method is fully described in Algorithm 1, in which H^t is the neighborhood function that satisfies $H^t : \mathbb{R}^+ \to \mathbb{R}^+$, $H^t(0) = 1$ and $\lim_{z\to+\infty} H^t(z) = 0$, and $\mu(t)$ is a training parameter. Generally, H^t and $\mu(t)$ are supposed to be decreasing with the iteration number, t.

Algorithm 1 Stochastic SOM algorithm for numeric datasets

Require: Numeric dataset: $\mathbf{X} = (x_{ij})_{i=1,\dots,n,\, j=1,\dots,d}$

1: For all $u = 1, \dots, U$ initialize p_u^0 randomly in $\mathbb{R}^d$
2: **for** $t = 1, \dots, T$ **do**
3: Randomly choose an input $i \in \{1, \dots, n\}$
4: *Assignment step*: find the unit of the prototype closest to $\mathbf{x}_i = (x_{i1}, \dots, x_{id})$:

$$f^t(\mathbf{x}_i) \leftarrow \arg\min_{u=1,\dots,U} \|\mathbf{x}_i - p_u^{t-1}\|^2$$

5: *Representation step*: $\forall\, u = 1, \dots, U$,

$$p_u^t \leftarrow p_u^{t-1} + \mu(t) H^t(d(f^t(\mathbf{x}_i), u)) \left(\mathbf{x}_i - p_u^{t-1}\right)$$

where $\mathbf{1}_i$ is a vector with a single non null coefficient at the *ith* position, equal to one.
6: **end for**
7: **return** Clustering: $(f^T(\mathbf{x}_i))_{i=1,\dots,n}$ and prototypes: $(p_u^T)_{u=1,\dots,U}$.

In **SOMbrero**, the following options can be passed to the function `trainSOM` that trains a SOM:

- the grid of the SOM is a square grid with $U = m_1 \times m_2$ units (m_1 and m_2 chosen by the user). Units are positioned in $\mathbb{R}^2$ at $(k_1(u), k_2(u))_{k_1(u)=1,\dots,m_1,\, k_2(u)=1,\dots,m_2}$. Default values are approximately $m_1 = m_2 = \sqrt{\frac{n}{10}}$ (with a minimum of 5 and a maximum of 10 for default values);
- the distance between units on the grid, d, is Euclidean (default) or any type of distance available in the R function `dist` or also a relationship denoted by `letremy` which corresponds to the original implementation by Patrick Letremy and switches between Euclidean distance and the distance "maximum" which is the maximum between the distance on the first and the second coordinates;
- the function H^t, also called neighborhood relationship, can be chosen as Gaussian (default), $H^t(z) = e^{-z^2/r(t)}$ in which $r(t)$ is decreasing with t, or can be of type `letremy`: $H^t(z) = \begin{cases} 0 & \text{if } z \leq r(t) \\ 1 & \text{otherwise} \end{cases}$, with $r(t)$ decreasing during the training.
- prototypes can be initialized randomly as described in step 1 of Algorithm 1 or to one observation of the dataset, randomly chosen, each or by positioning them regularly on the first to PC of a PCA performed on $\mathbf{X}$ or finally, to values specified by the user;
- $\mathbf{X}$ can be preprocessed by centering and scaling to unit variance or not preprocessed at all. In the case of a preprocessing of the data, the prototypes are returned in the original scale of $\mathbf{X}$;
- the assignment step can be performed in a standard way, as described in step 4 of Algorithm 1, or using the Heskes's modified assignment step [11]:

$$f^t(\mathbf{x}_i) \leftarrow \arg\min_{u=1,\dots,U} \sum_{u'=1}^{U} H^t(d(u, u'))\|\mathbf{x}_i - p_{u'}^{t-1}\|^2;$$

- the number of iterations T can be set by the user. Default values is equal to $5n$;
- finally, the user can chose to save intermediate states of the training, which includes saving the prototypes, the clustering for all observations and the value of

$$\mathcal{E}^t = \sum_{i=1}^{n} \sum_{u=1}^{U} H^t(d(f^t(\mathbf{x}_i), u))\|\mathbf{x}_i - p_u^t\|^2.$$

2.2 Plots and diagnosis functions

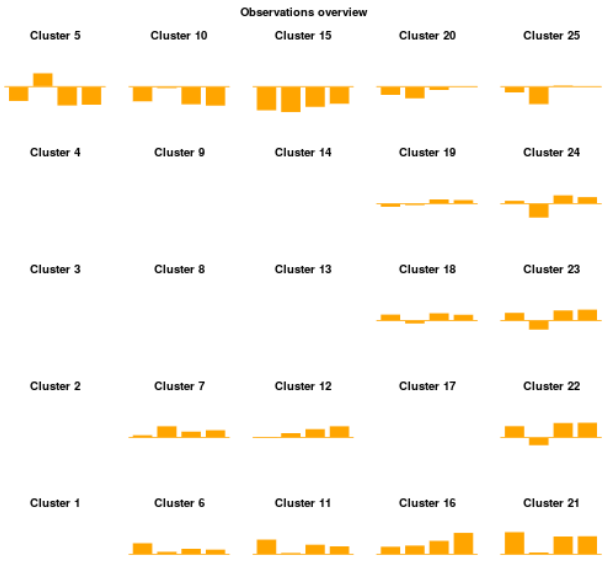
SOMbrero has been conceived to help the user interpret the output of the algorithm. More precisely, two standard quality measures [12] are given with the function `quality` that computes:

- the topographic error, which is the average frequency (over all observations) with which the prototype that comes second closest to an observation is not in the direct neighborhood (on the grid) of the winner prototype. It is a real number between 0 and 1, a value close to 0 indicating good quality;
- the quantization error, computed as:

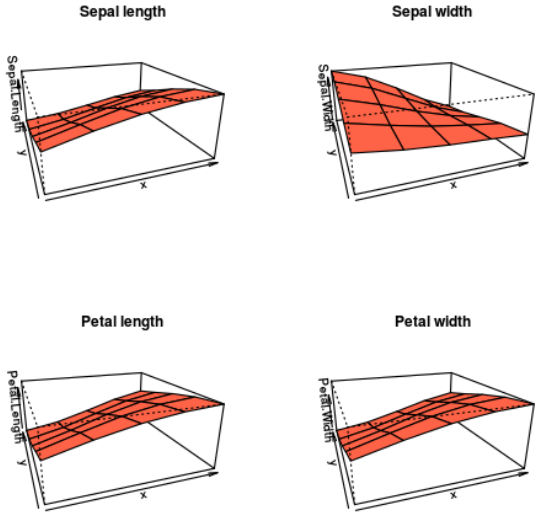
$$\mathcal{Q} = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{x}_i - p_{f(\mathbf{x}_i)}\|^2.$$

Moreover, **SOMbrero** comes with many different plots, that can be used to represent the original dataset, the prototypes or an additional variable related to the observations, on the grid. 16 types of plots are available with the single function `plot` and two options `what` (what to plot? observations, prototypes or additional information) and `type` (which type of graph to plot?). Some of these plots display distances between prototypes on the grid and can be used as diagnosis plots. An exhaustive description of available graphics is provided in [10] and examples of such plots, obtained on the famous iris dataset [13], are given in Figure 1.

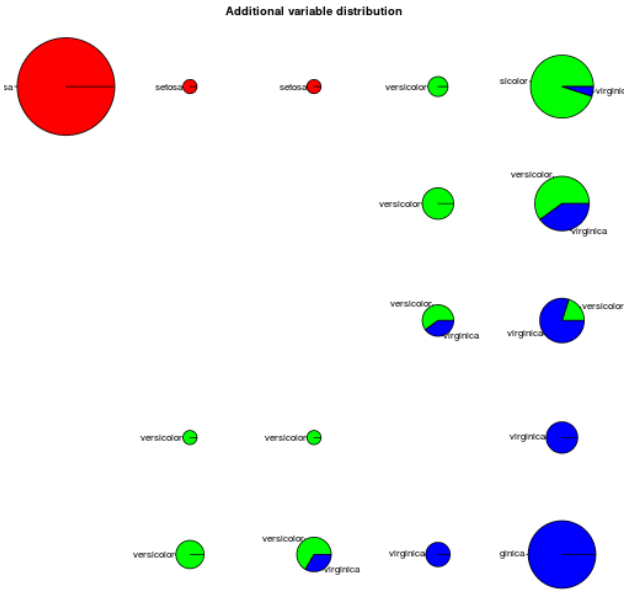
Finally, super-clusters can be obtained by using the function `superClass` on the output of the `trainSOM` function. This method performs hierarchical clustering on the prototypes of the map to cluster the units in "super-clusters".



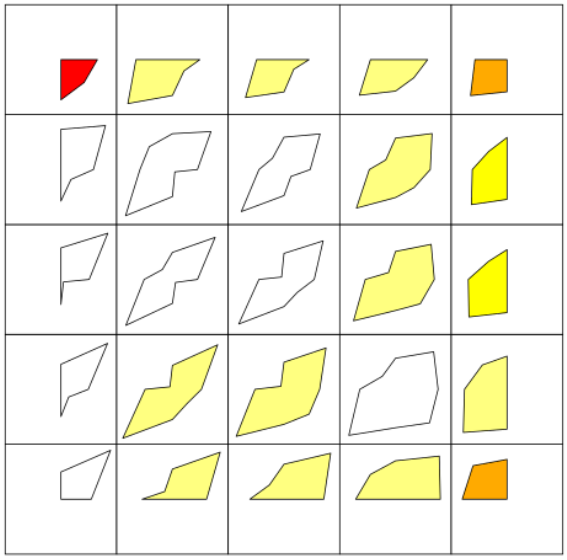
(a) `what="obs"` and `type="boxplot"`



(b) `what="prototypes"` and `type="3d"`



(c) `what="add"` and `type="pie"`



(d) `what="prototypes"` and `type="poly"`

Fig. 1: Different types of graphics obtained for the iris dataset: distribution of the variables within units (1a), prototype values for the different dimension (1b), additional variable (Species) distribution within units (1c) and distances between prototypes (1d) as proposed in [14].

These super-clusters can also been visualized on the map in various ways.

2.3 Documentation and graphical user interface

Additionally to the standard R documentation, **SOMbrero** also provides five vignettes (documentation files accessible from within the package and at https://cran.r-project.org/web/packages/SOMbrero/). The vignettes illustrate the functions with three datasets that correspond to the three variants implemented in the package. The first dataset is the iris dataset [13]. The second dataset (KORRESP variant, see Section 3) is the contingency table of votes for the combinations of "Départements" and "Candidates" during the French presidential election of 2002[4]. It provided in the package under the name `presidentielles2002`. The last dataset (relational variant, see Section 4) is the shortest path length between all nodes in the co-occurrence network obtained from the French novel "Les misérables" (V. Hugo) [15]. This dataset (co-occurrence network and dissimilarity matrix) is provided in the package under the name `lesmis`.

For users who are not familiar with R command lines, **SOMbrero** contains a graphical user interface which has

4. Source: http://www.interieur.gouv.fr/Elections/Les-resultats/Presidentielles

been programmed with the R package **shiny** [16] (see Figure 2). The GUI is launched from within R with the function `sombreroGUI()` and opens in default web browser.

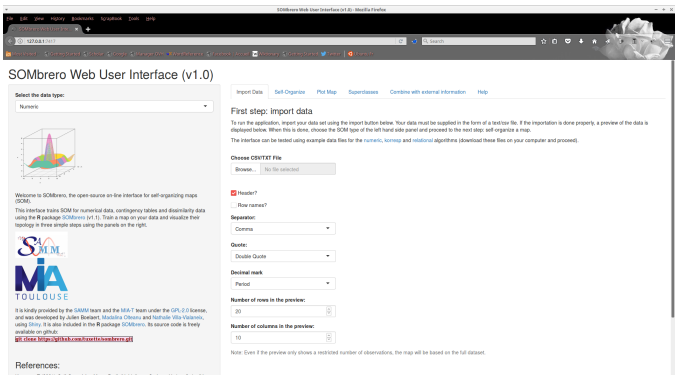


Fig. 2: Screenshot of the **SOMbrero** web user interface.

3 SOM FOR CONTINGENCY TABLES

Another very standard type of data that can be handled by **SOMbrero** is the case of contingency tables in which the dataset $\mathbf{T} = (n_{ij})_{i=1,\dots,p,j=1,\dots,q}$ is composed of joint frequencies for a pair of categorical variables (p is the number of levels for the first categorical variable and q the number of levels for the second). Classical correspondence analysis performs a weighted principal components analysis, using the χ^2 distance simultaneously on the row profiles and on the column profiles. The same principle is used in the so-called "KORRESP" algorithm, [17], [18], which extends SOM to contingency tables and is implemented in **SOMbrero**.

More precisely, the contingency table is first transformed into a dataset with row and column profiles:

$\mathbf{X} =$

	columns	rows
	row profile	*augmented row profile* — rows
	augmented column profile	*column profile* — columns

for which:

- row profiles, $(\mathbf{x}_i)_{i=1,\dots,p}$ are defined by $\forall j = 1,\dots,q$, $x_{ij} = \frac{n_{ij}}{n_{i.}} \times \sqrt{\frac{n}{n_{.j}}}$ and column profiles, $(\mathbf{x}_i)_{i=p+1,\dots,p+q}$ are defined similarly by $\forall j = q+1,\dots,q+p$, $x_{ij} = \frac{n_{j-q,i-p}}{n_{.,i-p}} \times \sqrt{\frac{n}{n_{j-q,.}}}$;
- augmented row profiles correspond to the most probable column profile for that row: $(\tilde{\mathbf{x}}_i)_{i=1,\dots,p}$ are defined by $\forall j = q+1,\dots,q+p$, $\tilde{x}_{ij} = x_{k(i)+p,j}$ with $k(i) = \arg\max_{k=1,\dots,q} x_{ik}$. Similarly, augmented column profiles are $(\tilde{\mathbf{x}}_i)_{i=p+1,\dots,p+q}$ with $\forall j = 1,\dots,q$, $\tilde{x}_{ij} = x_{k(i)-q,j}$ and $k(i) = \arg\max_{k=q+1,\dots,q+p} x_{ik}$.

The algorithm is fully described in Algorithm 2.

Prototypes are defined in $\mathbb{R}^{p+q}$. The algorithm then alternatively processes a row and a column (randomly chosen during the stochastic training process). For the chosen row (or column),

- the assignment step uses the profile $(\mathbf{x}_i)_{i=1,\dots,p}$ (in $\mathbb{R}^q$ for rows) or $(\mathbf{x}_i)_{i=p+1,\dots,p+q}$ (in $\mathbb{R}^p$ for columns). For a row $i \in \{1,\dots,p\}$, this gives:
$$f^t(\mathbf{x}_i) \leftarrow \arg\min_{u=1,\dots,U} \|\mathbf{x}_i - (p_{uj}^{t-1})_{j=1,\dots,q}\|^2;$$
- the representation step uses the augmented profile (in $\mathbb{R}^{p+q}$). For the chosen row i, this gives:
$$p_u^t \leftarrow p_u^{t-1} + \mu(t) H^t(d(f^t(\mathbf{x}_i), u)) \left(\begin{bmatrix} \mathbf{x}_i \\ \tilde{\mathbf{x}}_i \end{bmatrix} - p_u^{t-1} \right).$$

Algorithm 2 KORRESP variant of the stochastic SOM algorithm for contingency tables

Require: Contingency table: $\mathbf{T} = (n_{ij})_{i=1,\dots,p,\, j=1,\dots,q}$

1: Define $\mathbf{X}$ with row/column profiles and augmented row/column profiles
2: For all $u = 1,\dots,U$ initialize p_u^0 randomly in $\mathbb{R}^{p+q}$
3: **for** $t = 1,\dots,T$ **do**
4: **Start** *Process rows*
5: Randomly choose an input $i \in \{1,\dots,p\}$
6: *Assignment step*: find the unit of the prototype closest to $\mathbf{x}_i$:
$$f^t(\mathbf{x}_i) \leftarrow \arg\min_{u=1,\dots,U} \|\mathbf{x}_i - (p_{uj}^{t-1})_{j=1,\dots,q}\|^2$$
7: *Representation step*: $\forall\, u = 1,\dots,U$,
$$p_u^t \leftarrow p_u^{t-1} + \mu(t) H^t(d(f^t(\mathbf{x}_i), u)) \left(\begin{bmatrix} \mathbf{x}_i \\ \tilde{\mathbf{x}}_i \end{bmatrix} - p_u^{t-1} \right)$$
where $\mathbf{1}_i$ is a vector with a single non null coefficient at the *ith* position, equal to one.
8: **End**
9: **Start** *Process columns*
10: Randomly choose an input $i \in \{p+1,\dots,p+q\}$
11: *Assignment step*: find the unit of the prototype closest to $\mathbf{x}_i$:
$$f^t(\mathbf{x}_i) \leftarrow \arg\min_{u=1,\dots,U} \|\mathbf{x}_i - (p_{uj}^{t-1})_{j=q+1,\dots,q+p}\|^2$$
12: *Representation step*: $\forall\, u = 1,\dots,U$,
$$p_u^t \leftarrow p_u^{t-1} + \mu(t) H^t(d(f^t(\mathbf{x}_i), u)) \left(\begin{bmatrix} \tilde{\mathbf{x}}_i \\ \mathbf{x}_i \end{bmatrix} - p_u^{t-1} \right)$$
where $\mathbf{1}_i$ is a vector with a single non null coefficient at the *ith* position, equal to one.
13: **End**
14: **end for**
15: **return** Clustering: $(f^T(\mathbf{x}_i))_{i=1,\dots,p+q}$ and prototypes: $(p_u^T)_{u=1,\dots,U}$.

This method can be used in **SOMbrero** with `trainSOM(..., type= "korresp")` and provides a map in which levels for both variables of the contingency tables are clustered simultaneously. This approach is analogous to the common representation provided for contingency tables in correspondence analysis.

4 SOM FOR RELATIONAL DATASETS

4.1 Relational SOM

In the case where the observations $(x_i)_{i=1,\dots,n}$ take values in an arbitrary input space $\mathcal{G}$, a widely used representation of the data is to compute a measures of dissemblance or resemblance between pairs of observations. Well known examples of such frameworks include shortest path lengths between pairs of nodes in a graph [19], string edit distance between categorical time series [20] or DNA samples [21] or various kernels used in many application fields, including biology [22]. In this section, we will consider that the data are described by a dissimilarity measure $\mathbf{\Delta} = (\delta_{ij})_{i,j=1,\dots,n}$, which is such that $\delta_{ij} = \delta(x_i, x_j)$ is the measure of dissemblance between x_i and x_j, is symmetric and null on the diagonal. Note that a natural Euclidean structure is not necessarily associated with this dissimilarity measure.

Several extensions of the SOM algorithm have been proposed in this context, including "median SOM" [23]–[25] and kernel SOM [26]–[28]. **SOMbrero** implements the relational variant of SOM [5], [21] which relies to a pseudo-Euclidean framework. More precisely, [29] shows that, given δ as described above, we can find two Euclidean spaces, $\mathcal{E}_+$ and $\mathcal{E}_-$, and two mappings $\phi^+ : \mathcal{G} \to \mathcal{E}_+$ and $\phi^- : \mathcal{G} \to \mathcal{E}_-$ such that

$$\delta(x, x') = \|\phi^+(x) - \phi^+(x')\|^2_{\mathcal{E}_+} - \|\phi^-(x) - \phi^-(x')\|^2_{\mathcal{E}_-}.$$

Writing $\phi(x) := (\phi^+(x), \phi^-(x))$, that take values in the orthogonal direct sum of the two Euclidean spaces implicitely defined by the dissimilarity, the prototypes can thus be defined as $p_u = \sum_{i=1}^n \beta_{ui}\phi(x_i)$ with $\beta_{ui} \geq 0$ and $\sum_{i=1}^n \beta_{ui} = 1$. Using the standard operations in the pseudo-Euclidean space induced by the similarity, the representation and assignment steps of the SOM algorithm can be rewritten as described in Algorithm 3 (justifications are given in [21]).

Algorithm 3 Relational variant of the stochastic SOM algorithm for dissimilarity data

1: For all $u = 1, \dots, U$ and $i = 1, \dots, n$, initialize β_{ui}^0 such that $\beta_{ui}^0 \geq 0$ and $\sum_i^n \beta_{ui}^0 = 1$.
2: **for** $t = 1, \dots, T$ **do**
3: Randomly choose an input x_i
4: *Assignment step*: find the unit of the closest prototype

$$f^t(x_i) \leftarrow \arg\min_{u=1,\dots,U} \left(\left(\beta_u^{t-1}\mathbf{\Delta}\right)_i - \frac{1}{2}(\beta_u^{t-1})^T \mathbf{\Delta} \beta_u^{t-1} \right)$$

where $\left(\beta_u^{t-1}\mathbf{\Delta}\right)_i$ is the *ith* entry of $\beta_u^{t-1}\mathbf{\Delta}$
5: *Representation step*: $\forall\, u = 1, \dots, U$,

$$\beta_u^t \leftarrow \beta_u^{t-1} + \mu(t) H^t(d(f^t(x_i), u)) \left(\mathbf{1}_i - \beta_u^{t-1}\right)$$

where $\mathbf{1}_i$ is a vector with a single non null coefficient at the *ith* position, equal to one.
6: **end for**

If the dissimilarity matrix is a Euclidean distance, then the relational SOM is exactly identical to the standard numerical SOM as long as the prototypes of the original SOM are initialized in the convex hull of the original data (*i.e.*, the initial prototypes can be written $p_u^0 = \sum_i \beta_{ui}^0 x_i$). Similarly, the relational SOM is identical to kernel SOM as described in [27], [30], [31] for a dissimilarity defined from a kernel K by

$$\delta(x_i, x_j) := K(x_i, x_i) + K(x_j, x_j) - 2K(x_i, x_j).$$

4.2 Acceleration of relational SOM

However, the complexity of the assignment and representation steps of stochastic relational SOM are, respectively, $\mathcal{O}(n^2U)$ and $\mathcal{O}(nU)$, as pointed in [32]. This leads to a total complexity of $\mathcal{O}(n^2U)$ for one iteration. To obtain good convergence properties, the algorithm requires a number of iterations of the order of αn, as shown in [33], yielding a complexity of $\mathcal{O}(\alpha n^3 U)$. Hence, relational SOM is not adapted to large datasets and cannot be used to analyze more than a few thousands observations. [34] have proposed two approximate versions to overcome this issue, using sparse representations of the prototypes or DR preprocessing techniques. However, the version implemented in **SOMbrero** uses the solution proposed in [35] which yields to a complexity of $\mathcal{O}(\alpha n^2 U)$.

More precisely, the method is based on a re-formulation of the assignment step (step 4 in Algorithm 3):

$$f^t(x_i) = \arg\min_{u=1,\dots,U} B_u^{t-1} - \frac{1}{2} A_{ui}^{t-1}$$

in which

$$A^t = \left(\sum_{j,j'=1}^n \beta_{uj}^t \beta_{uj'}^t \delta_{jj'} \right)_{u=1,\dots,U}$$

is a vector of size U and

$$B^t = \left(\sum_{j=1}^n \beta_{uj}^t \delta_{i'j} \right)_{u=1,\dots,U, i'=1,\dots,n}$$

is a $(U \times n)$-matrix.

The updates of A^t and B^t are performed during the representation step, which is thus equivalent to $\forall u = 1, \dots, U$, $\beta_u^t = (1 - \lambda_u(t))\beta_u^{t-1} + \lambda_u(t)\mathbf{1}_i$, in which $\lambda_u(t) = \mu(t)H^t(d(f^t(x_i), u))$. This leads to the following updates:

$$B_{ui'}^t = \sum_{j=1}^n \beta_{uj}^t \delta_{i'j} = (1 - \lambda_u(t)) B_{ui'}^{t-1} + \lambda_u(t)\delta_{i'i},$$

and

$$\begin{aligned} A_u^t &= \sum_{j,j'=1}^n \beta_{uj}^t \beta_{uj'}^t \delta_{jj'} \\ &= (1 - \lambda_u(t))^2 A_u^{t-1} + \lambda_u(t)^2 \delta_{ii} \\ &\quad + 2\lambda_u(t)(1 - \lambda_u(t)) B_{ui}^{t-1}. \end{aligned}$$

Provided that the assignment and representation steps are usually performed $\mathcal{O}(\alpha n)$ times, the total complexity of the algorithm is dominated by $\mathcal{O}(\alpha n^2 U)$. This computational cost is obtained using the additional storage of A^t and B^t which requires an additional memory of $\mathcal{O}(U)$ and $\mathcal{O}(nU)$, respectively. This solution is implemented in **SOMbrero** since version 1.2, which considerably reduced the computational time required by the relational version, as shown in [35].

4.3 Special features for graphs

As already mentionned, graphs are a special case of relational data: if $(x_i)_i$ are the nodes of a given graphs, several types of similarities/dissimilarities can be used to describe ressemblance between those nodes. Widely used examples include shortest path lengths or kernels based on the Laplacian of the graph [36] that include, among others, the commute time kernel, $K_{\mathrm{CT}} = L^+$ [37] or the heat kernel, $K_{\mathrm{H}} = e^{-\beta L}$ ($\beta > 0$), [38].

SOMbrero includes additional functions for this type of data. When the clustered entities are nodes of a graph $\mathcal{G}$, a projection of the graph onto the map can be obtained with the function `projectIGraph`. This projected graph has a number of nodes equal to the number of (nonempty) units in the map and edges that connect pairs of vertices that contain each at least one node connected by an edge in $\mathcal{G}$. The output of this function is given as an `igraph`[5] that can provide a simplified representation of the graph as in Figure 3 (this figure provides a simplified representation of the graph `lesmis` described in Section 2.3 after a superclustering has been applied to the result of the relational SOM algorithm).

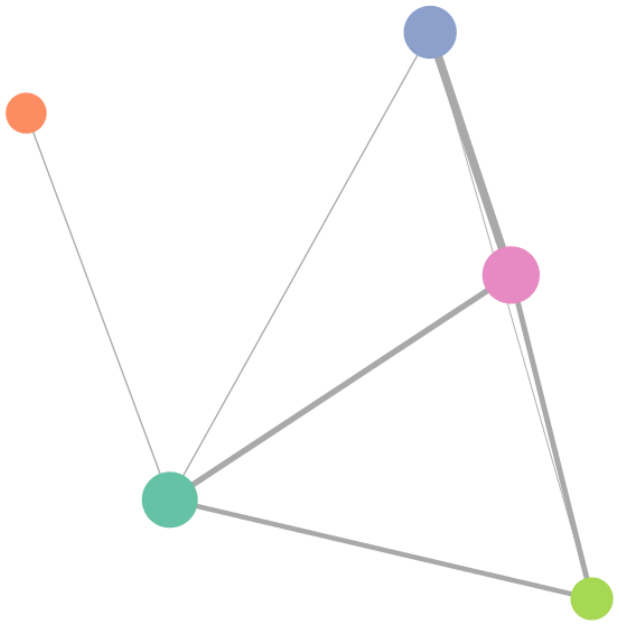
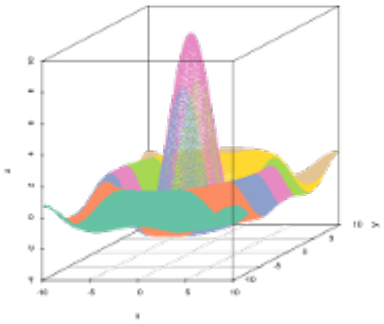


Fig. 3: Simplified representation of the co-appearance network of "Les misérables" as obtained with **SOMbrero**.

Acknowledgment

This paper summarizes articles written with Marie Cottrell, Jérôme Mariette, Madalina Olteanu, Fabrice Rossi, Laura Bendhaïba and Julien Boelaert, who have all contributed to the development of **SOMbrero**.

5. available from the R package **igraph**, [39].



References

[1] T. Kohonen, *Self-Organizing Maps, 3rd Edition*. Berlin, Heidelberg, New York: Springer, 2001, vol. 30.

[2] ——, *MATLAB Implementations and Applications of the Self-Organizing Map*. Helsinki, Finland: Unigrafia Oy, 2014.

[3] ——, "Analysis of a simple self-organizing process," *Biological Cybernectis*, vol. 44, pp. 135–140, 1982.

[4] ——, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, pp. 59–69, 1982.

[5] B. Hammer and A. Hasenfuss, "Topographic mapping of large dissimilarity data sets," *Neural Computation*, vol. 22, no. 9, pp. 2229–2284, September 2010.

[6] C. Bishop, M. Svensén, and C. Williams, "GTM: The generative topographic mapping," *Neural Computation*, vol. 10, no. 1, pp. 215–234, 1998.

[7] T. Kohonen, "Learning vector quantization," in *The Handbook of Brain Theory and Neural Networks*, M. Arbib, Ed. Cambridge, MA, USA: MIT Press, 1995, pp. 537–540.

[8] P. Schneider, B. Hammer, and M. Biehl, "Adaptive relevance matrices in learning vector quantization," *Neural Computation*, vol. 21, pp. 3532–3561, 2009.

[9] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2017. [Online]. Available: http://www.R-project.org

[10] J. Boelaert, L. Bendhaïba, M. Olteanu, and N. Villa-Vialaneix, "SOMbrero: an r package for numeric and non-numeric self-organizing maps," in *Advances in Self-Organizing Maps and Learning Vector Quantization (Proceedings of WSOM 2014)*, ser. Advances in Intelligent Systems and Computing, T. Villmann, F. Schleif, M. Kaden, and M. Lange, Eds., vol. 295. Mittweida, Germany: Springer Verlag, Berlin, Heidelberg, 2014, pp. 219–228.

[11] T. Heskes, "Energy functions for self-organizing maps," in *Kohonen Maps*, E. Oja and S. Kaski, Eds. Amsterdam: Elsevier, 1999, pp. 303–315. [Online]. Available: http://www.snn.ru.nl/reports/Heskes.wsom.ps.gz

[12] G. Pölzlbauer, "Survey and comparison of quality measures for self-organizing maps," in *Proceedings of the Fifth Workshop on Data Analysis (WDA'04)*, J. Paralic, G. Pölzlbauer, and A. Rauber, Eds. Sliezsky dom, Vysoke Tatry, Slovakia: Elfa Academic Press, 2004, pp. 67–82.

[13] R. Becker, J. Chambers, and A. Wilks, *The New S Language*. Wadsworth & Brooks/Cole, 1988.

[14] M. Cottrell and E. de Bodt, "A Kohonen map representation to avoid misleading interpretations," in *Proceedings of the European Symposium on Artificial Neural Networks*, M. Verleysen, Ed. Bruxelles, Belgium: Editions D Facto, 1996, pp. 103–110.

[15] D. Knuth, *The Stanford GraphBase: A Platform for Combinatorial Computing*. Reading, MA: Addison-Wesley, 1993.

[16] RStudio and Inc., *shiny: Web Application Framework for R*, 2013, R package version 0.6.0. [Online]. Available: http://CRAN.R-project.org/package=shiny

[17] M. Cottrell, P. Letrémy, and E. Roy, "Analyzing a contingency table with Kohonen maps: a factorial correspondence analysis," in *Proceedings of International Workshop on Artificial Neural Networks (IWANN 93)*, ser. Lecture Notes in Computer Science, J. Cabestany, J. Mary, and A. E. Prieto, Eds. Springer Verlag, 1993, pp. 305–311.

[18] M. Cottrell and P. Letrémy, "How to use the Kohonen algorithm to simultaneously analyse individuals in a survey," *Neurocomputing*, vol. 63, pp. 193–207, 2005.

[19] M. Olteanu and N. Villa-Vialaneix, "Using SOMbrero for clustering and visualizing graphs," *Journal de la Société Française de Statistique*, vol. 156, no. 3, pp. 95–119, 2015. [Online]. Available: http://publications-sfds.math.cnrs.fr/index.php/J-SFdS/article/view/473

[20] S. Massoni, M. Olteanu, and N. Villa-Vialaneix, "Which distance use when extracting typologies in sequence analysis? An application to school to work transitions," in *International Work Conference on Artificial Neural Networks (IWANN 2013)*, Puerto de la Cruz, Tenerife, 2013.

[21] M. Olteanu and N. Villa-Vialaneix, "On-line relational and multiple relational SOM," *Neurocomputing*, vol. 147, pp. 15–30, 2015.

[22] B. Schölkopf, K. Tsuda, and J. Vert, *Kernel Methods in Computational Biology*. London, UK: MIT Press, 2004.

[23] T. Kohohen and P. Somervuo, "Self-organizing maps of symbol strings," *Neurocomputing*, vol. 21, pp. 19–30, 1998.

[24] B. Conan-Guez, F. Rossi, and A. El Golli, "Fast algorithm and implementation of dissimilarity self-organizing maps," *Neural Networks*, vol. 19, no. 6-7, pp. 855–863, 2006.

[25] A. El Golli, F. Rossi, B. Conan-Guez, and Y. Lechevallier, "Une adaptation des cartes auto-organisatrices pour des données décrites par un tableau de dissimilarités," *Revue de Statistique Appliquée*, vol. LIV, no. 3, pp. 33–64, 2006.

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

[27] D. Mac Donald and C. Fyfe, "The kernel self organising map." in *Proceedings of 4th International Conference on knowledge-based Intelligence Engineering Systems and Applied Technologies*, 2000, pp. 317–320.

[28] K. Lau, H. Yin, and S. Hubbard, "Kernel self-organising maps for classification," *Neurocomputing*, vol. 69, pp. 2033–2040, 2006.

[29] L. Goldfarb, "A unified approach to pattern recognition," *Pattern Recognition*, vol. 17, no. 5, pp. 575–582, 1984.

[30] P. Andras, "Kernel-Kohonen networks," *International Journal of Neural Systems*, vol. 12, pp. 117–135, 2002.

[31] N. Villa and F. Rossi, "A comparison between dissimilarity SOM and kernel SOM for clustering the vertices of a graph," in *6th International Workshop on Self-Organizing Maps (WSOM 2007)*. Bielefield, Germany: Neuroinformatics Group, Bielefield University, 2007.

[32] F. Rossi, "How many dissimilarity/kernel self organizing map variants do we need?" in *Advances in Self-Organizing Maps and Learning Vector Quantization (Proceedings of WSOM 2014)*, ser. Advances in Intelligent Systems and Computing, T. Villmann, F. Schleif, M. Kaden, and M. Lange, Eds., vol. 295. Mittweida, Germany: Springer Verlag, Berlin, Heidelberg, 2014, pp. 3–23.

[33] M. Olteanu, N. Villa-Vialaneix, and M. Cottrell, "On-line relational SOM for dissimilarity data," in *Advances in Self-Organizing Maps (Proceedings of WSOM 2012)*, ser. AISC (Advances in Intelligent Systems and Computing), P. Estévez, J. Príncipe, P. Zegers, and G. Barreto, Eds., vol. 198. Santiago, Chile: Springer Verlag, Berlin, Heidelberg, 2013, pp. 13–22.

[34] J. Mariette, M. Olteanu, and N. Villa-Vialaneix, "Efficient interpretable variants of online SOM for large dissimilarity data," *Neurocomputing*, vol. 225, pp. 31–48, 2017.

[35] J. Mariette, F. Rossi, M. Olteanu, and N. Villa-Vialaneix, "Accelerating stochastic kernel som," in *XXVth European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2017)*, M. Verleysen, Ed. Bruges, Belgium: d-side publications, 2017, pp. 269–274.

[36] A. Smola and R. Kondor, "Kernels and regularization on graphs," in *Proceedings of the Conference on Learning Theory (COLT) and Kernel Workshop*, ser. Lecture Notes in Computer Science, M. Warmuth and B. Schölkopf, Eds. Washington, DC, USA: Springer-Verlag Berlin Heidelberg, 2003, pp. 144–158.

[37] F. Fouss, A. Pirotte, J. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 3, pp. 355–369, 2007.

[38] R. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," in *Proceedings of the 19th International Conference on Machine Learning*, C. Sammut and A. Hoffmann, Eds. Sydney, Australia: Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 2002, pp. 315–322.

[39] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *InterJournal*, vol. Complex Systems, 2006. [Online]. Available: http://igraph.sf.net