SOMbrero: An R Package for Numeric and Non-numeric Self-Organizing Maps

Julien Boelaert[1], Laura Bendhaiba[1],
Madalina Olteanu[1], and Nathalie Villa-Vialaneix[1,2,3]

[1] SAMM, Université Paris 1 Panthéon-Sorbonne,
90 rue de Tolbiac, 75634 Paris cedex 13, France
[2] UPVD, 52 avenue Paul Alduy, 66860 Perpignan cedex 9, France
[3] INRA, Unité MIAT, BP 52627 31326 Castanet Tolosan cedex, France
{julien.boelaert,laurabendhaiba}@gmail.com,
madalina.olteanu@univ-paris1.fr, nathalie.villa@toulouse.inra.fr

Abstract. This paper presents **SOMbrero**, a new R package for self-organizing maps. Along with the standard SOM algorithm for numeric data, it implements self-organizing maps for contingency tables ("Korresp") and for dissimilarity data ("relational SOM"), all relying on stochastic (i.e., on-line) training. It offers many graphical outputs and diagnostic tools, and comes with a user-friendly web graphical interface, based on the **shiny** R package.

Keywords: Self-Organizing Maps, R, Dissimilarity, Korresp.

1 Introduction

Self-Organizing Maps (SOM), introduced by Teuvo Kohonen [1], are a popular clustering and visualization algorithm. While originally intended for data consisting exclusively of numeric vectors, this prototype-based learning algorithm has been extended to handle other types of data.

One of the oldest attempts to generalize the SOM algorithm to non numeric data is the so-called "Korresp" algorithm (and related methods [2]), that extends standard correspondence analysis: this approach can be used to cluster rows and columns of a contingency table (i.e., values of two categorical variables that are jointly observed) on a topological map. More recently, several extensions of the SOM algorithm to non numeric data have been proposed. The median principle has been used to handle data described by dissimilarity matrices: [3] uses it by replacing the standard computation of prototypes by an approximation in the original data set. Since this approach is very restrictive, it has been improved in [4,5,6] for data described by a kernel, in [7,8] for data described by a dissimilarity matrix and in [9,8] for data described by several dissimilarity matrices or several kernels. In these works, the prototypes are no longer numeric vectors of the input space as in classical SOM, but convex combinations of the observations in a Hilbert or a pseudo-Euclidean vector space. The dissimilarity versions are called "relational SOM" and the kernel versions "kernel SOM". It should be noted

T. Villmann et al. (eds.), *Advances in Self-Organizing Maps and Learning Vector Quantization,* Advances in Intelligent Systems and Computing 295,
DOI: 10.1007/978-3-319-07695-9_21,

that relational SOM is a generalization of kernel SOM when the dissimilarity that describes the data is not Euclidean [8].

Several implementations of the SOM algorithm exist in different mathematical/statistical softwares, most on them usable only for numeric data. The SOM Toolbox[1] is a Matlab library implementing many variants of SOM for numeric data, with graphical outputs, user interfaces and implementations of other clustering algorithms. It is partially based on the original implementation `SOM_PAK` [10]. SAS Entreprise Miner (current version 12.3) features an implementation of SOM for numeric data. Also, the SAS programs by Patrick Letremy[2] implement standard SOM and several extensions, including SOM for contingency tables as described in [2].

R is one of the most popular statistical software environments, and several R packages implement variants of the SOM algorithm:

- **class** (current version 7.3-9, last updated in August 2013) offers a crude implementation of the SOM algorithm for numeric data with batch training;
- **som** [11] (current version 0.3-5, last updated in April 2010) implements a two-step batch algorithm for numeric data, with basic plotting of the resulting map;
- **popsom** [12] (current version 2.3, last updated in October 2013) is built on the **som** package, with additional diagnostic tools and visualizations;
- **kohonen** [13] (current version 2.0.14, last updated in December 2013) implements the standard SOM for numeric data as well as "super-organized maps", in which the observed variables can be separated into distinct "layers", and two versions of supervised SOM, X-Y fused SOM and Bi-Directional Kohonen maps, in which class information is available for all observations in addition to numeric coordinates. The training is done stochastically and several plot options are available;
- **yasomi** [14] (current version 0.3, last updated in March 2011) implements batch algorithms for standard SOM, relational SOM and kernel SOM (for data consisting of pairwise evaluations of a positive semi-definite kernel function). It features data driven construction methods for the maps, and several plotting options.

In the present article, we describe a new R package for SOM: **SOMbrero**. It implements stochastic versions of the SOM algorithms for numeric data, dissimilarity data, and for data described by contingency tables. To our knowledge, **SOMbrero** is also the R package that proposes the largest number of diagnostic tools (graphics, super-classes and quality measures) designed to help the user understand the outputs of the algorithm. The package uses the S3 object-oriented standard. Its current version (version 0.4-1, last updated in November 2013) is available on R-Forge at `http://sombrero.r-forge.r-project.org/`. It runs

[1] Current version 2.1, last updated in 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`

on R 3.0 or higher, depends on packages including **wordcloud**, **scatterplot3d**, **igraph** and **e1071**, and comes with a **shiny** web graphical user interface (version 0.1) that can be tested at `http://shiny.nathalievilla.org/sombrero/` or directly loading the package into R and running the command:

```
sombreroGUI()
```

The rest of this article is organized as follows : Section 2 presents the main features of **SOMbrero**. Section 3 describes the web graphical user interface and Section 4 gives short examples of applications to various types of data.

2 Main Features of SOMbrero

This section describes the main features of **SOMbrero**: the kind of data the package is able to handle, the available plots and the other diagnostics tools.

Three Types of Self-Organizing Maps: The **SOMbrero** package currently supports three types of stochastic training algorithms, each designed for a specific type of data : "standard" SOM for numeric data, "Korresp" for contingency tables, and "relational SOM" for dissimilarity data.

Numeric SOM. When the data are numeric vectors, **SOMbrero** uses the standard stochastic SOM algorithm [1], which iterates over two steps. In the *affectation step*, a single observation is randomly drawn and affected to the one prototype it is closest to (in terms of Euclidean distance, or some other chosen distance, in the input space). The *representation step* then updates all prototypes, moving the closest unit and its neighbors towards the drawn observation. The algorithm converges empirically towards a minimum of the extended within-class variance. In the particular case of a finite data set, [15] showed that the SOM algorithm trained with a fixed neighborhood radius is equivalent to a gradient-descent minimizing a cost function equal to the extended within-class variance. The stochastic version of the algorithm is preferred over the batch version as it generally provides a better organization, whatever the initialization, at a comparable computational cost [16].

Korresp. When the data consist of a contingency table for two categorical variables, 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 Korresp algorithm [2], which extends SOM to contingency tables.

Relational SOM. SOMbrero also implements a stochastic version of relational SOM, described in [8], which is an extension of SOM to dissimilarity data. This is useful when the data are not naturally described by a fixed set of numerical attributes (e.g. categorical variables or relations between objects), but when a measure of resemblance between observations (i.e., a similarity or a dissimilarity) can nevertheless be constructed. In this approach, prototypes are expressed as a symbolic convex combination of the observations that is justified by a pseudo-Euclidean framework [7].

Graphical Outputs: The main advantage of SOM over other clustering algorithms is that it combines clustering with a nonlinear projection since the clusters are organized on a grid, while preserving the topology of the original data. **SOMbrero** offers a wide range of plots aimed at giving a comprehensive overview of the resulting clusters.

All available plots are obtained using a single function `plot.somRes` (or `plot.somSC` if the visualization must be combined with a super-clustering as described below) and just two arguments handle the type of the output plot in a handy way: `what` and `type`. Argument `what` must be one of: `obs` for plotting a graphic based on the original observations, `prototypes` for plotting a graphic related to the prototypes, `energy` for plotting the evolution of the extended within-class variance during training and `add` for combining the clustering with one or several additional variables. Argument `type` sets which type of graphic is to be plotted (colors, pie charts, bars, distances...). Table 1 summarizes all available graphics for the three implemented SOM algorithms.

Table 1. Summary of plots available in **SOMbrero** 0.4-1

type	numeric			korresp			relational		
	obs	proto	add	obs	proto	add	obs	proto	add
color	x	x	x		x				x
3d		x			x				
lines	x	x	x		x			x	x
barplot	x	x	x		x			x	x
radar	x	x	x		x			x	x
boxplot	x		x						x
poly.dist		x			x			x	
umatrix		x			x			x	
smooth.dist		x			x			x	
mds		x			x			x	
grid.dist		x			x			x	
hitmap	x			x			x		
names	x		x	x			x		x
words			x						x
pie			x						x
graph			x						x

Plot types fall into two main categories: they either display the distances between prototypes (only when `what="prototypes"`), or the actual values, for each cluster, of the prototypes, observations or additional variables (for `what="prototypes"`, `what="obs"` or `what="add"`, respectively).

The plots that display the values (values of the prototypes, or average values of the observations or additional variables) are listed below:

- `color` shows the value of a single variable using a gradient of colors;
- `3d` is similar to `color` but shows the values as a three-dimensional surface plot;

- `lines` shows the values of all variables together, with lines;
- `barplot` is similar to `lines` but uses vertical bars and is limited to 5 variables at most;
- `radar` is similar to `lines` but uses radar plots;
- `boxplot` is similar to `lines` but uses boxplots and is limited to 5 variables at most. This option is used when the user wants to visualize the distribution of the variables and not only their means.

The plots that display distances between prototypes are listed below:

- `poly.dist` represents the distances between neighboring prototypes with polygons plotted for each cell of the grid [17]. The smaller the distance between a polygon's vertex and a cell border, the closer the pair of prototypes. The polygons are filled with colors indicating the number of observations in each cell;
- `umatrix` is the well known "u-matrix" [18] that plots the grid and fills the cells with colors according to the mean distance between a prototype and the neighboring prototypes;
- `smooth.dist` depicts the average distance between a prototype and its neighbors using smooth color changes;
- `mds` plots a two-dimensional Multi Dimensional Scaling projection of the prototypes;
- `grid.dist` plots all two-way grid distances (computed on the grid) against the corresponding prototype distances (computed in the input space).

Other Plots Are also Available: `type="hitmap"` displays the distribution of the observations on the map with rectangles; each cluster is represented by a rectangle with an area proportional to the number of observations it contains, [19]. Using `type="names"` or `type="words"` (the latter only available for additional variables) displays a grid of word clouds, either of observation names or of words related to the observations of each cluster. Finally, `type="pie"` or `type="graph"` can be used to display pie charts (for an additional categorical variable) or graphs (with nodes corresponding to the observations that have been clustered on the map).

Diagnostic Tools: **SOMbrero** offers additional diagnostic tools: super-clustering (the result of which can be plotted on most graphical outputs) and quality measures. The SOM algorithm often results in a large number of classes, which is not very handy for interpretation. Therefore, a common practice is to run an ascending hierarchical clustering algorithm on the prototypes of the trained map. **SOMbrero** implements this in function `superClass`. A dendrogram and a scree-plot can be drawn using function `plot.somSC`, which can guide the user's choice.

Furthermore, apart from the energy plots, **SOMbrero** offers two measures of quality for a trained SOM, through function `quality` [20]:

- 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, which is the distance (in the input space) of observations to their assigned prototype, averaged over all observations. It is a positive real number, with a value close to 0 when projection quality is good.

3 Graphical User Interface

SOMbrero comes with a user-friendly graphical interface, which makes most of its options available in a few clicks, without resorting to the command line. The interface is programmed using the R package **shiny** [21]. It can be tested using a simple web browser (some of the features may not work with Internet Explorer or Chrome; Firefox must be preferred), and can be accessed on-line at `http://shiny.nathalievilla.org/sombrero` or in R using the command

```
sombreroGUI()
```

It is shown in Figure 1.

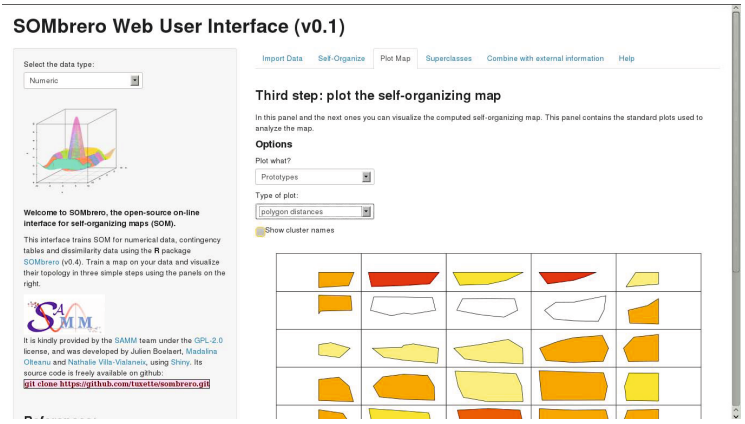

Fig. 1. Screenshot of the **SOMbrero** web user interface

The interface consists of seven panels: the left hand side panel allows the user to choose the type of SOM and gives general information and references. An "Import Data" panel is used to import a data file in csv or text format, and to set formatting options for the importation. If the data are properly imported, a preview table is shown in this panel. The "Self-Organize" panel is used to select the SOM options and train the algorithm. The "Plot Map" panel provides the different graphical outputs implemented in **SOMbrero**. The "Superclasses" panel is used to compute and to display super-classes. The "Combine with external information" panel can be used to import additional data and to display them on the map and thus to combine the results of SOM with external information. Finally, the "Help" panel contains indications about how to use the interface.

4 Examples

SOMbrero also provides five vignettes (documentation files accessible from within the package) that detail the use of the package for different types of data. Three datasets, provided with **SOMbrero**, are used to illustrate the numeric case, the Korresp case and the relational case. The present section provides a few examples of the analyses that can be performed using **SOMbrero**; we refer the reader to the package's vignettes for comprehensive illustrations.

Numeric SOM: the `iris` data set. The numeric SOM case is illustrated with the Fisher's famous *iris* dataset [22]. The following two command lines are used to train the SOM and assess the quality of the map[3]:

```
iris.som <- trainSOM(iris[,1:4])
quality(iris.som)
# $topographic          $quantization
# [1] 0.06              [1] 0.1933871
```

The results of the algorithm can be combined with the categorical variable `Species`, plotted as pie-charts in Figure 2. This graphic shows a good organization and separation of the different species on the map.

```
plot(iris.som, what= "add", variable= iris$Species)
```

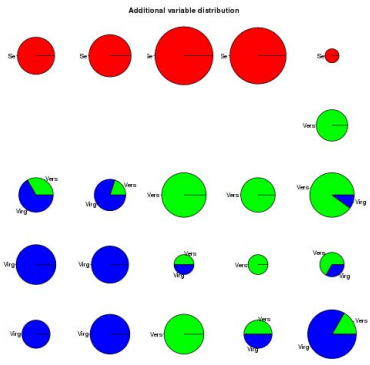

Fig. 2. Pie charts of `Species` for a SOM trained on the four numeric variables of the *iris* dataset

Korresp: French Presidential Election Data. The second example illustrates the Korresp algorithm using the `presidentielles2002` data set[4]. The data consist of a contingency table containing the number of votes for each region

[3] The values of these quality measures may vary between maps because of the stochastic nature of SOM training procedure.

[4] Source: "Ministère de l'Intérieur, France", `http://www.interieur.gouv.fr/content/download/1789/18734/file/Presidentielle-2002-departements.zip`

(département, in rows) and each candidate (columns) in the first round of the 2002 French presidential elections.

The `names` plot shows the codes of the départements and the names of the candidates in their assigned cells, as in Figure 3. Departments with similar vote results are plotted close together, along with the candidates most representative of their voting profiles:

```
presi.som <- trainSOM(presidentielles2002,
                      type= "korresp")
plot(presi.som, what= "obs", type= "names")
```

The top left side of the map corresponds to the extreme right candidates, and corresponding "départements", that attracted much attention during these elections. More information about the interpretation of these results is given in the package's corresponding vignette.

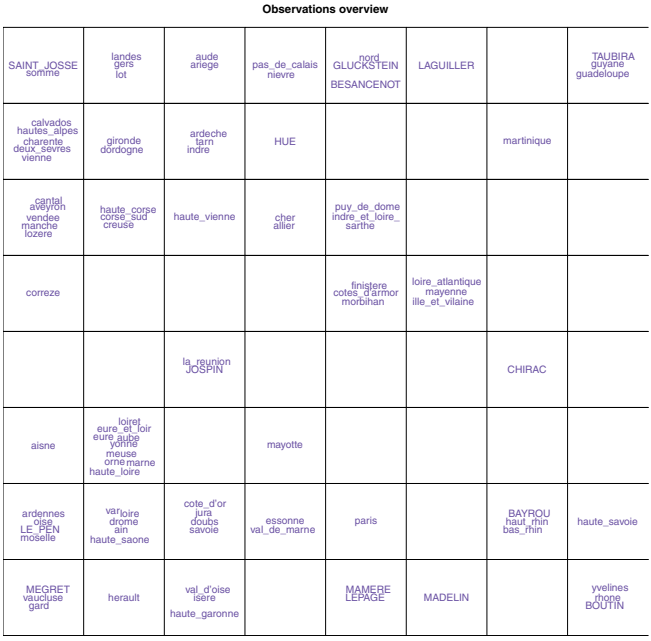

Fig. 3. Names plot for a Korresp analysis of French presidential elections

Relational SOM: Data Set "Les Misérables". Data set `lesmis`[5], is the co-appearance network of the characters in Victor Hugo's novel "Les Misérables". A dissimilarity matrix has been derived from this graph (using the shortest path lengths), which can be used as input for relational SOM.

Figure 4 shows a projection of the original graph of characters onto the SOM grid: each node in this figure represents a cluster and has an area proportional

[5] Source: `http://people.sc.fsu.edu/~jburkardt/datasets/sgb/jean.dat`

to the number of characters classified inside. The edge widths are proportional to the number of connections between the characters of the two clusters. Colors highlight superclasses computed by hierarchical clustering on the SOM prototypes. The SOM clustering, the super classes and the graph of Figure 4 are obtained with just three command lines:

```
lesmis.som <-trainSOM(dissim.lesmis,type="relational")
lesmis.SC <- superClass(lesmis.som, k=6)
plot(lesmis.SC, what="add", type="graph", var=lesmis)
```

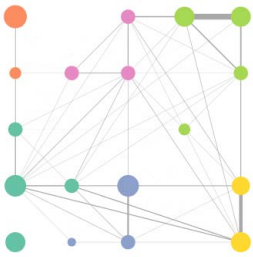

Fig. 4. Projection of the graph of characters' links on a relational SOM grid

5 Conclusion

The **SOMbrero** R package implements on-line algorithms of SOM for three types of data (numeric vectors, contingency tables and dissimilarity data), and provides multiple diagnostic tools, graphical outputs, as well as a handy graphical user interface. Further versions intend to include support for other types of data (multiple categorical data, multiple dissimilarity SOM), and to add training options, such as different grid topologies and weighting of the observations.

References

1. Kohonen, T.: Self-Organizing Maps, 3rd edn., vol. 30. Springer, Heidelberg (2001)
2. Cottrell, M., Letremy, P., Roy, E.: Analyzing a contingency table with Kohonen maps: a factorial correspondence analysis. In: Mira, J., Cabestany, J., Prieto, A.G. (eds.) IWANN 1993. LNCS, vol. 686, pp. 305–311. Springer, Heidelberg (1993)
3. Kohohen, T., Somervuo, P.: Self-organizing maps of symbol strings. Neurocomputing 21, 19–30 (1998)
4. Mac Donald, D., Fyfe, C.: The kernel self organising map. In: Proceedings of 4th International Conference on Knowledge-Based Intelligence Engineering Systems and Applied Technologies, pp. 317–320 (2000)
5. Andras, P.: Kernel-Kohonen networks. International Journal of Neural Systems 12, 117–135 (2002)

6. Villa, N., Rossi, F.: A comparison between dissimilarity SOM and kernel SOM for clustering the vertices of a graph. In: 6th International Workshop on Self-Organizing Maps (WSOM), Bielefeld, Germany, Neuroinformatics Group, Bielefeld University (2007)
7. Hammer, B., Hasenfuss, A.: Topographic mapping of large dissimilarity data sets no access. Neural Computation 22(9), 2229–2284 (2010)
8. Olteanu, M., Villa-Vialaneix, N.: On-line relational and multiple relational som. Neurocomputing (forthcoming, 2014)
9. Olteanu, M., Villa-Vialaneix, N., Cierco-Ayrolles, C.: Multiple kernel self-organizing maps. In: Verleysen, M. (ed.) XXIst European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Bruges, Belgium, pp. 83–88. d-side publications (2013)
10. Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J.: Som_pak: The self-organizing map program package. Technical Report A31, Helsinki University of Technology, Laboratory of Computer and Information Science (1996)
11. Yan, J.: som: Self-Organizing Map. R package version 0.3-5 (2010)
12. Hamel, L., Ott, B., Breard, G.: popsom: Self-Organizing Maps With Population Based Convergence Criterion. R package version 2.3 (2013)
13. Wehrens, R., Buydens, L.: Self- and super-organising maps in r: the kohonen package. J. Stat. Softw. 21(5) (2007)
14. Rossi, F.: yasomi: Yet Another Self Organising Map Implementation. R package version 0.3/r39 (2012)
15. Ritter, H., Martinetz, T., Shulten, K.: Neural computation and Self-Organizing Maps, an Introduction. Addison-Wesley (1992)
16. Fort, J., Letremy, P., Cottrell, M.: Advantages and drawbacks of the batch kohonen algorithm. In: Verleysen, M. (ed.) Proceedings of 10th European Symposium on Artificial Neural Networks (ESANN 2002), Bruges, Belgium, pp. 223–230 (2002)
17. Cottrell, M., de Bodt, E.: A Kohonen map representations to avoid misleading interpretations. In: Verleysen, M. (ed.) Proceedings of ESANN 1996, D Facto, Bruxelles, pp. 103–110 (1996)
18. Ultsch, A., Siemon, H.: Kohonen's self organizing feature maps for exploratory data analysis. In: Proceedings of International Neural Network Conference, INNC 1990 (1990)
19. Vesanto, J.: Data Exploration Process Based on the Self–Organizing Map. PhD thesis, Helsinki University of Technology, Espoo (Finland), Acta Polytechnica Scandinavica, Mathematics and Computing Series No.115 (2002)
20. Polzlbauer, G.: Survey and comparison of quality measures for self-organizing maps. In: Paralic, J., Polzlbauer, G., Rauber, A. (eds.) Proceedings of the Fifth Workshop on Data Analysis (WDA 2004), Sliezsky dom, Vysoke Tatry, Slovakia, pp. 67–82. Elfa Academic Press (2004)
21. RStudio, Inc.: shiny: Web Application Framework for R. R package version 0.6.0 (2013)
22. Becker, R., Chambers, J., Wilks, A.: The New S Language. Wadsworth & Brooks/Cole (1988)