Knowledge Discovery by Accuracy Maximization

Stefano Cacciatore , Leonardo Tenori, Claudio Luchinat, Phillip R. Bennet and David A. MacIntyre

2022-08-31

1 Introduction

KODAMA (Knowledge Discovery by Accuracy Maximization) is a novel learning algorithm for unsupervised feature extraction, specifically designed for analysing noisy and high-dimensional data sets (Cacciatore *et al.*, 2014).The core idea of the original algorithm is to use an iteration procedure to produce a hypothetical classification through maximization of cross-validated predictive accuracy. Using only the data set as input (no *a priori* knowledge is needed), an iterative procedure permits classification with a high cross-validated accuracy. Two different classifiers are implemented in this package for the computation of cross-validated predictive accuracy: k-nearest neighbors (kNN) and Partial Least Squares - Discriminant Analysis (PLS-DA). This procedure is repeated several times to average the effects owing to the randomness of the iterative procedure. After each run of the procedure, a classification vector with high cross-validated accuracy is obtained. KODAMA subsequently collects and processes these results by constructing a dissimilarity matrix to provide a holistic view of the data. This documentation introduces the usage of KODAMA.

2 Installation

2.1 Installation via CRAN

The R package KODAMA (current version 1.1) is part of the Comprehensive R Archive Network (CRAN)[1]. The simplest way to install the package is to enter the following command into your R session: `install.packages("KODAMA")`. We suggest to install the R package `rgl` for the data visualization in 3D interactive plots.

2.2 Manual installation from source

To compile the C/C++ code included in the package for customization or installation on alternative operating systems the package can be manually installed from source. To this end, open the package's page at CRAN (Cacciatore *et al.*, 2014) and then proceed as follows:

- Download KODAMA.tar.gz and save it to your hard disk
- Open a shell/terminal/command prompt window and change to the desired directory for installation of KODAMA.tar.gz. Enter `R CMD INSTALL KODAMA.tar.gz` to install the package. Note that this may require additional software on some platforms. Windows requires Rtools[2] to be installed and to be available in the default search path (environment variable PATH). MAC OS X requires installation of Xcode developers and command line tools.

[1] https://cran.r-project.org/
[2] https://developer.apple.com/xcode/

2.3 Compatibility issues

All versions downloadable from CRAN have been built using R version, R.3.2.3. The package should work without major issues on R versions $> 3.0.0$.

3 Getting Started

To load the package, enter the following instruction in your R session:

```
## Loading required package: minerva
```

```
## Loading required package: Rtsne
```

If this command terminates without any error messages, you can be sure that the package has been installed successfully. The KODAMA package is now ready for use.

The package includes both a user manual (this document) and a reference manual (help pages for each function). To view the user manual, enter `vignette("KODAMA")`. Help pages can be viewed using the help command `help(package="KODAMA")`.

4 Datasets

4.1 Swiss Roll

The function `swissroll` computes the Swiss Roll dataset of a given sample size. The following example computes a Swiss Roll dataset containing 1,000 samples.

```
require("rgl")
x=swissroll()
open3d()
plot3d(x, col=rainbow(1000),box=FALSE,type="s",size=1)
```

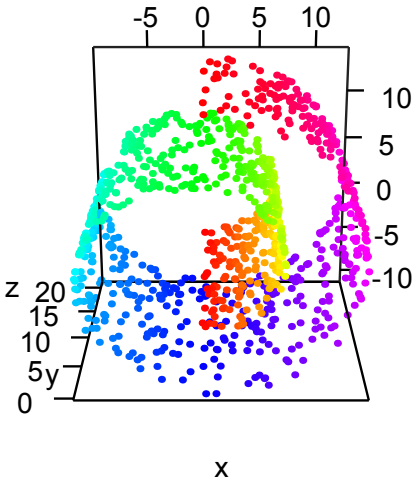

Figure 1: Three dimensional Swiss Roll dataset.

4.2 Ulisse Dini's surface

The function `dinisurface` computes the Ulisse Dini's surface dataset of a given sample size. The following example computes a Ulisse Dini's surface dataset containing 1,000 samples.

```
require("rgl")
x=dinisurface()
open3d()
plot3d(x, col=rainbow(1000),box=FALSE,type="s",size=1)
```

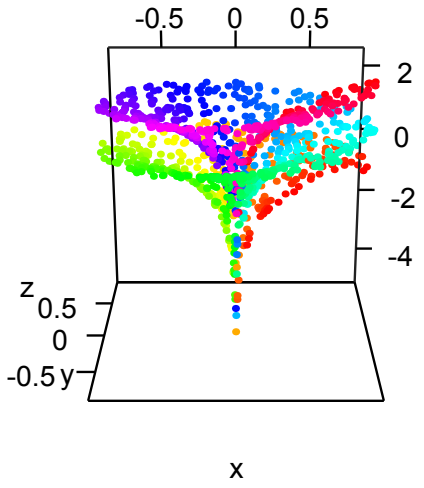

Figure 2: Three dimensional Ulisse Dini's surface dataset.

4.3 Helicoid

The function `helicoid` computes the Helicoid dataset of a given sample size. The following example computes a Helicoid dataset containing 1,000 samples.

```
require("rgl")
x=helicoid()
open3d()
plot3d(x, col=rainbow(1000),box=FALSE,type="s",size=1)
```

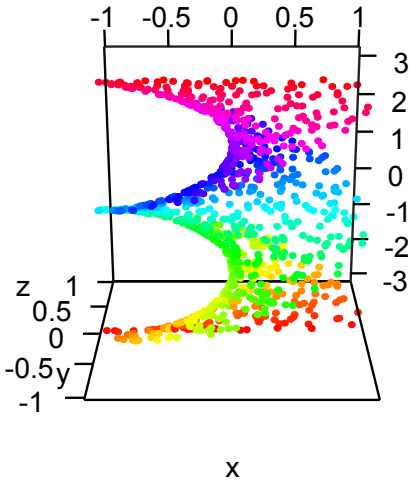

Figure 3: Three dimensional Helicoid dataset.

4.4 Spirals

The function `spirals` computes the Spirals dataset of a given sample size. The following example computes a Spirals dataset containing 1,000 samples.

```
par(mfrow=c(2,2),mai=c(0,0,0,0))
v1=spirals(c(100,100,100),c(0.1,0.1,0.1))
plot(v1,col=rep(2:4,each=100))
v2=spirals(c(100,100,100),c(0.1,0.2,0.3))
plot(v2,col=rep(2:4,each=100))
v3=spirals(c(100,100,100,100,100),c(0,0,0.2,0,0))
plot(v3,col=rep(2:6,each=100))
v4=spirals(c(20,40,60,80,100),c(0.1,0.1,0.1,0.1,0.1))
plot(v4,col=rep(2:6,c(20,40,60,80,100)))
```

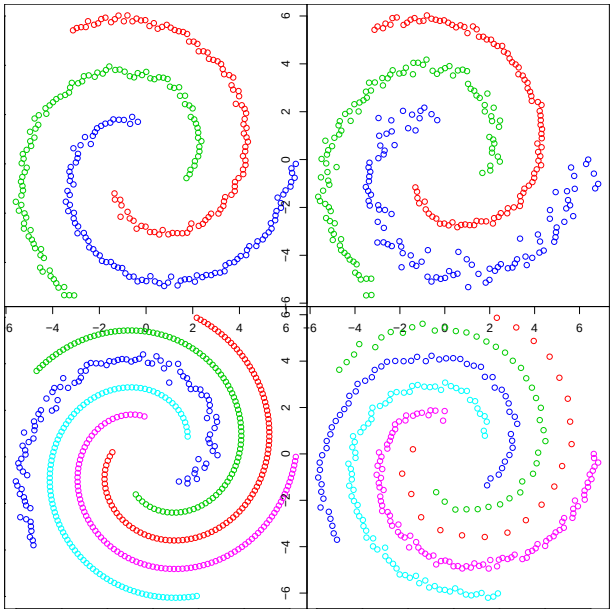

Figure 4: Four different two dimensional Spirals datasets.

4.5 Lymphoma

The `lymphoma` dataset consists of gene expression profiles of the three most prevalent adult lymphoid malignancies: diffuse large B-cell lymphoma (DLBCL), follicular lymphoma (FL), and B-cell chronic lymphocytic leukemia (B-CLL). The dataset consists of 4,682 mRNA genes for 62 samples (42 samples of DLBCL, 9 samples of FL and 11 samples of B-CLL). Missing values are imputed and data are standardized as described in Dudoit, *et al.* (2002).

4.6 MetRef

The data of belong to a cohort of 22 healthy donors (11 male and 11 female) where each provided about 40 urine samples over the time course of approximately 2 months, for a total of 873 samples. Each sample was analyzed by Nuclear Magnetic Resonance Spectroscopy. Each spectrum was divided in 450 spectral bins.

4.7 State of the Union

The `USA` dataset consists of the spoken, not written, presidential addresses from 1900 until the sixth address by Barack Obama in 2014. Punctuation characters, numbers, words shorter than three characters, and stop-words (*e.g.*, "that", "and", and "which") were removed from the dataset. This resulted in a dataset of 86 speeches containing 834 different meaningful words each. Term frequency-inverse document frequency (TF-IDF) was used to obtain feature vectors. It is often used as a weighting factor in information retrieval and text mining. The TF-IDF value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to control for the fact that some words are generally more common than others.

4.8 Iris

This famous Fisher's (aka Anderson's) iris data set gives the measurements (centimetres) of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are *Iris setosa*, *I. versicolor*, and *I. virginica.* iris is a data frame with 150 cases (rows) and 5 variables (columns) named `Sepal.Length`, `Sepal.Width`, `Petal.Length`, `Petal.Width` and `Species`.

5 Starting with KODAMA

We suggest KODAMA is first tested with the Iris dataset using the default parameters (*i.e.*, kNN classifier with k=2). The KODAMA function automatically performs the fast T-distributed Stochastic Neighbor Embedding (van der Maaten 2014) on the KODAMA dissimilarity matrix. The results can be visualized with the `plot` function.

```
data(iris)
data=iris[,-5]
labels=iris[,5]
kodama_knn_5=KODAMA(data,FUN="KNN",f.par = 2)
plot(kodama_knn_2$scores,pch=21,bg=as.numeric(labels)+1, xlab="First component", ylab="Second component"
```

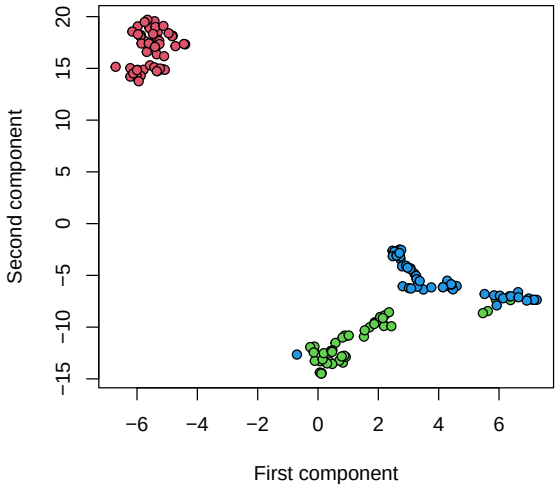

Figure 5: KODAMA on the Iris dataset.

5.2 Evaluation of the Monte Carlo iterative process

The `mcplot` function can be used to extract the values of cross-validated accuracy obtained from each iterative step of the Monte Carlo procedures of maximization of the cross-validated accuracy.

```
mcplot(kodama_knn_5)
```

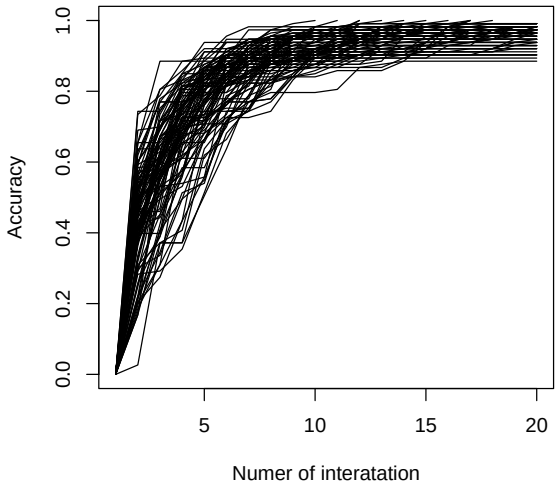

Figure 6: Cross-validated accuracy.

5.1 Adjusting Input Preferences

KODAMA can be run with different input settings. The two different classifiers, kNN and PLS-DA, are tested. The classifier and its parameter can be changed by modifying `FUN` and `f.par`, respectively. The kNN classifier is tested with k=2, 5, 10 and the PLS-DA classifier is tested with 2, 3 and 4 components.

```
kodama_knn_2 =KODAMA(data,FUN="KNN",f.par=2)
kodama_knn_5 =KODAMA(data,FUN="KNN",f.par=5)
kodama_knn_10=KODAMA(data,FUN="KNN",f.par=10)
kodama_pls_2 =KODAMA(data,FUN="PLS-DA",f.par=2)
kodama_pls_3 =KODAMA(data,FUN="PLS-DA",f.par=3)
kodama_pls_4 =KODAMA(data,FUN="PLS-DA",f.par=4)
```

After the KODAMA analyses, the different solutions can be visualized for comparative purposes.

```
par(mfrow=c(2,3))
plot(kodama_knn_2$scores ,pch=21,bg=as.numeric(labels)+1,xlab="First component",
                    ylab="Second component",main="KODAMA with kNN (k=2)")
plot(kodama_knn_5$scores ,pch=21,bg=as.numeric(labels)+1,xlab="First component",
                    ylab="Second component",main="KODAMA with kNN (k=5)")
plot(kodama_knn_10$scores,pch=21,bg=as.numeric(labels)+1,xlab="First component",
                    ylab="Second component",main="KODAMA with kNN (k=10)")
plot(kodama_pls_2$scores ,pch=21,bg=as.numeric(labels)+1,xlab="First component",
                    ylab="Second component",main="KODAMA with PLS (ncomp=2)")
plot(kodama_pls_3$scores ,pch=21,bg=as.numeric(labels)+1,xlab="First component",
                    ylab="Second component",main="KODAMA with PLS (ncomp=3)")
plot(kodama_pls_4$scores ,pch=21,bg=as.numeric(labels)+1,xlab="First component",
                    ylab="Second component",main="KODAMA with PLS (ncomp=4)")
```

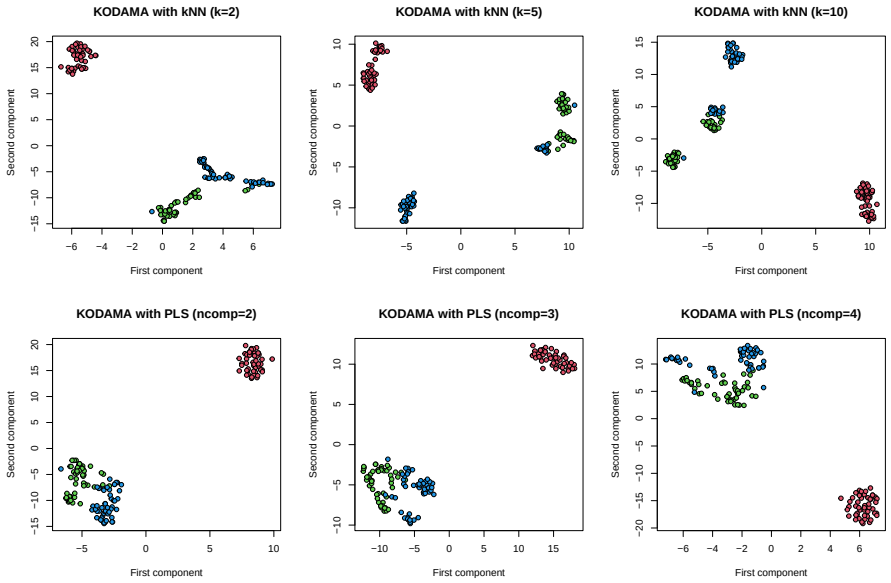

Figure 7: KODAMA results obtained with different input settings.

The Shannon Entropy (H) (Shannon, 1948), a measure of unpredictability of information content, can be used to choose the best classifier and optimize its relative parameters. H is given by:

$H = \sum_i \sum_j v_{ij} \times log\, v_{ij}$

where v_{ij} is the proximity value between the sample i and the sample j divided by the sum of overall proximities.

Different classifiers can lead to solutions that represent different interpretation of the data. A lower H value can be indicative of the solution that best characterizes the data.

Classifier	parameter	Entropy
kNN	2	9.371
kNN	5	9.362
kNN	10	9.381
PLS-DA	2	9.976
PLS-DA	3	9.933
PLS-DA	4	9.977

5.3 k-test

The `k.test` function performs a statistical test to assess association between the KODAMA output and any additional related parameters such as clinical metadata. The coefficient of determination (R^2) is used to assess the proportion of the variance in the dependent variable (KODAMA output) that is predictable from the independent variable and can be thus used as a measure of the goodness of fit (Cameron *et al.*, 1997). A permutation test is performed by randomly sampling the value of the labels to estimate the significance of the observed association.

5.2 Loadings

The `loads` function can be used to extract the variable ranking. After each maximization of the cross-validated accuracy the final label set is used to calculate the loadings of PLS-DA or the p-value from the Kruskal-Wallis Rank Sum test. The output of the `loads` function is the average of these values for each variable.

5.3 Unsupervised and semi-supervised

In the next example, MetRef dataset is used to show all possibilities offered by KODAMA performing the analysis in unsupervised or semi-supervised fashion. Firstly the MetRef dataset is pre-processed. This involves removing zero values from the MetRef dataset matrix and correcting for variations in urine concentration using Probabilistic Quotient Normalization (Dieterle *et al.* 2006).

```
data(MetRef)

# Zero values are removed.
MetRef$data=MetRef$data[,-which(colSums(MetRef$data)==0)]

# Normalization of the data set with the Probabilistic Quotient    Normalization method.
MetRef$data=normalization(MetRef$data)$newXtrain
# Centers the mean to zero and scales data by dividing each variable by the variance.
MetRef$data=scaling(MetRef$data)$newXtrain
donor=as.numeric(as.factor(MetRef$donor))
gender=as.numeric(as.factor(MetRef$gender))
```

KODAMA is performed using different input settings.

```
kk1=KODAMA(MetRef$data,FUN = "PLS-DA",f.par = 2)
kk2=KODAMA(MetRef$data,FUN = "PLS-DA",f.par = 5)
kk3=KODAMA(MetRef$data,FUN = "PLS-DA",f.par = 10)
kk4=KODAMA(MetRef$data,FUN = "PLS-DA",f.par = 20)
kk5=KODAMA(MetRef$data,FUN = "PLS-DA",f.par = 50)
kk6=KODAMA(MetRef$data,FUN = "PLS-DA",f.par = 100)
kk7=KODAMA(MetRef$data,FUN = "KNN",f.par = 2)
kk8=KODAMA(MetRef$data,FUN = "KNN",f.par = 3)
kk9=KODAMA(MetRef$data,FUN = "KNN",f.par = 5)
kk10=KODAMA(MetRef$data,FUN = "KNN",f.par = 10)
kk11=KODAMA(MetRef$data,FUN = "KNN",f.par = 15)
kk12=KODAMA(MetRef$data,FUN = "KNN",f.par = 20)
```

The most informative solution is represented by KODAMA using the PLS-DA with 100 components as the internal classifier as can be seen in the table below, which presents all Entropy values of each solution.

Classifier	parameter	Entropy
kNN	2	12.847
kNN	3	12.228
kNN	5	12.129
kNN	10	12.432
kNN	15	12.783
kNN	20	13.137
PLS-DA	2	13.37
PLS-DA	5	13.416
PLS-DA	10	13.322
PLS-DA	20	12.637
PLS-DA	50	11.327
PLS-DA	100	11.307

Principal Component Analysis, performed on this data set, is shown in the figure. Colors are indicative of donor specificity (left plot) and gender of donor (right plot).

```
par(mfrow=c(1,2))
plot(pca(MetRef$data)$x,
     bg=rainbow(22)[donor],pch=21,cex=1.5)
plot(pca(MetRef$data)$x,
     bg=c("#2c7ac8","#e3b80f")[gender],pch=21,cex=1.5)
```

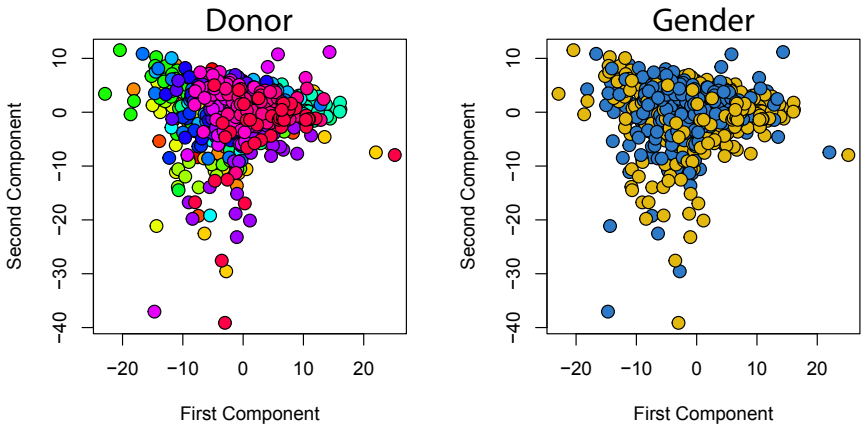

Figure 8: Principal Component Analysis.

The result of KODAMA with PLS-DA using 100 components as the classifier is subsequently shown. The KODAMA output saved in the variable pp is calculated by applying Sammon's Non-Linear Mapping (Sammon 1969) to the KODAMA dissimilarity matrix.

```
par(mfrow=c(1,2))
plot(kk6$scores,bg=rainbow(22)[donor],pch=21,
     xlab="First Component", ylab="Second Component",cex=1.5)
plot(kk6$scores,bg=c("#2c7ac8","#e3b80f")[gender],pch=21,
     xlab="First Component", ylab="Second Component",cex=1.5)
```

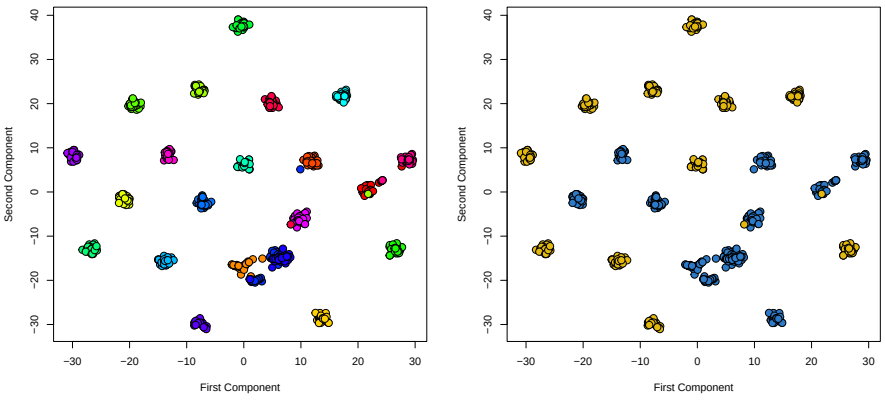

Figure 9: KODAMA with PLS-DA with 100 components as classifier.

KODAMA can also use external information to work in a semi-supervised way. Supervised constraints can be imposed by linking some samples in such a way that if one of them is changed the linked ones must change in the same way (*i.e.*, they are forced to belong to the same class). This will produce solutions where linked samples are forced to have the lowest values in the KODAMA dissimilarity matrix.

In the next example, urine samples from the same donor are linked by providing class (donor) information to the algorithm before performing the iterative procedure thereby undertaking a "semi-supervised approach" to highlight otherwise hidden features.

```
kkA=KODAMA(MetRef$data,FUN="PLS-DA",f.par = 100,constrain=donor )
par(mfrow=c(1,2))
plot(kkA$scores,bg=rainbow(22)[donor],pch=21,
     xlab="First Component", ylab="Second Component",cex=1.5)
plot(kkA$scores,bg=c("#2c7ac8","#e3b80f")[gender],pch=21,
     xlab="First Component", ylab="Second Component",cex=1.5)
```

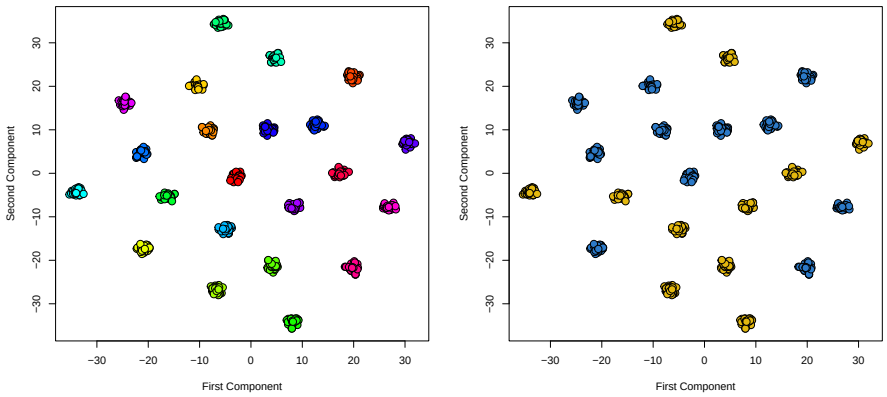

Figure 10: Semi-supervised KODAMA with constrain.

Additional external information can be provided through fixing labels of vector `W` on the KODAMA algorithm. The value of the vector `fix` must be `TRUE` or `FALSE`. By default all elements are `FALSE`. Samples with the `TRUE` fix value will not change the class label defined in W during the maximization of the cross-validated accuracy procedure.

Here, gender information for the first ten donors is provided. Color coding indicates gender in the figures. Square data points indicate samples with supervised information. Circle data points indicate samples without any supervised information.

```
FIX=sample(c(TRUE,FALSE),873,T)
inform=gender
inform[!FIX]=NA
kkB=KODAMA(MetRef$data,FUN="PLS-DA",f.par = 100,W=inform,fix=FIX)
par(mfrow=c(1,2))
plot(kkB$scores,bg=rainbow(22)[donor],pch=21,
     xlab="First Component", ylab="Second Component",cex=1.5)
plot(kkB$scores,bg=c("#2c7ac8","#e3b80f")[gender],pch=21,
     xlab="First Component", ylab="Second Component",cex=1.5)
```

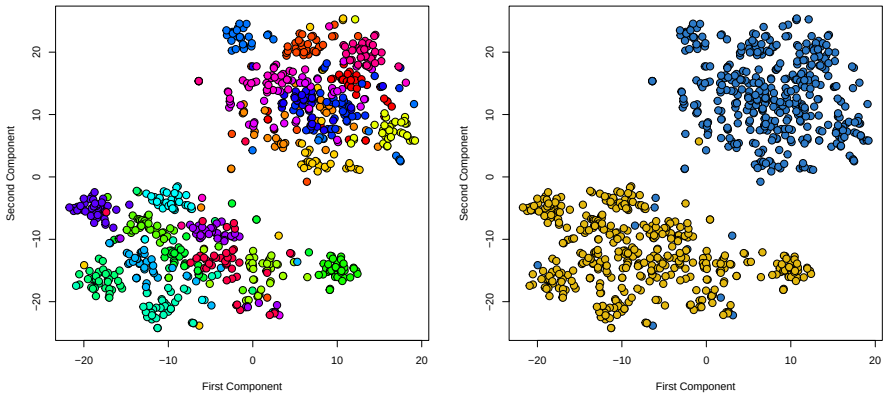

Figure 11: Semi-supervised KODAMA with fixed samples.

Here, information of the last two examples are provided together.

```
FIX=donor>10
inform=gender
inform[!FIX]=NA
kkC=KODAMA(MetRef$data,FUN="PLS-DA",f.par = 100,W= inform,constrain=donor,fix=FIX)
par(mfrow=c(1,2))
plot(kkC$scores,bg=rainbow(22)[donor],pch=21,
     xlab="First Component", ylab="Second Component",cex=1.5)
plot(kkC$scores,bg=c("#2c7ac8","#e3b80f")[gender],pch=21,
     xlab="First Component", ylab="Second Component",cex=1.5)
```

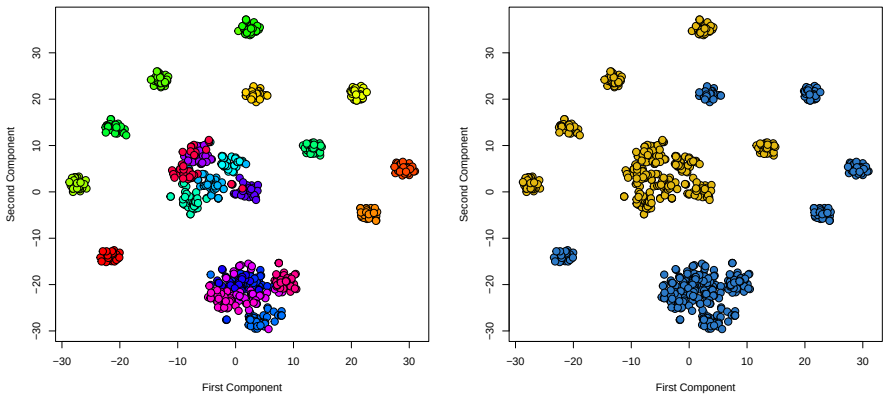

Figure 12: Semi-supervised KODAMA with fixed samples and constrain.

6 Special Notes for Users Upgrading from Previous Version

Version 2.0 has introduced a few novelty to the KODAMA algorithm and some small changes in the output structure of the functions. We tried to ensure backward compatibility with the previous version where possible. Sammon's Non-Linear Mapping (Sammon 1969) used in the previous version to transform the KODAMA dissimilarity matrix in a space dimensionality has been replaced by t-SNE (van der Maaten 2008) with Tree-Based implementation (van der Maaten 2014) using the Rtsne package.

Version 1.1 has brought several fundamental changes to the architecture of the package. We tried to ensure backward compatibility with the previous version where possible. However, there are still some caveats the users should take into account.

Users who upgrade to version 1.1 from the older version (0.0.1) should be aware that the package requires the new version of `Rcpp` and `RcppArmadillo` packages. This issue can simply be solved by installing `Rcpp` and `RcppArmadillo` from CRAN using `install.packages("Rcpp")` and `install.packages("RcppArmadillo")`.

7 Errors and Solutions

7.1 Rcpp, RcppArmadillo and OS X Mavericks "-lgfortran" and "-lquadmath" error

Compiling the source R package, we reported the following error:

```
ld: warning: directory not found for option '-L/usr/local/lib/gcc/x86_64-apple-darwin13.0.0/4.8.2'
ld: library not found for -lquadmath
clang: error: linker command failed with exit code 1 (use -v to see invocation)
make: *** [KODAMA.so] Error 1
ERROR: compilation failed for package 'KODAMA'
* removing '/Users/dmacinty/Library/R/3.3/library/KODAMA'
```

Mainly, R for OS X Maverick was compiled using gfortran-4.8. The Solution is to go to the optional libraries, frameworks and applications for Mac OS X on r.research.att.com and download gfortran-4.8.2-darwin13.tar.bz2. Extract the package in ~/, which is root. The files should be unpacked into /usr/local/...

Alternatively, open terminal and type:

```
curl -O http://r.research.att.com/libs/gfortran-4.8.2-darwin13.tar.bz2
sudo tar fvxz gfortran-4.8.2-darwin13.tar.bz2 -C /
```

8 How to Cite this Package

Cacciatore S, Tenori L, Luchinat C, Bennett P, and MacIntyre DA. KODAMA: an updated R package for knowledge discovery and data mining. Bioinformatics. Submitted.

Moreover, the original paper in which KODAMA was introduced should also be cited as follows:

Cacciatore S, Luchinat C, Tenori L. Knowledge discovery by accuracy maximization. Proc Natl Acad Sci U S A 2014;111(14):5117-22.

To obtain BibTex entries of the two references, you can enter the following into your R session to Bibtex `citation("KODAMA")`.

9 References

Cacciatore S, Luchinat C, Tenori L (2014) Knowledge discovery by accuracy maximization. *Proc Natl Acad Sci USA*,111, 5117-22.

Alizadeh AA, *et al.* (2000) Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769), 503-11.

Cameron CA, et al. (1997) An R-squared measure of goodness of fit of some common nonlinear regression models. *J Econometrics* 77(2), 1790-2.

Dieterle, F et al. (2006) Probabilistic Quotient Normalization as Robust Method to Account for Diluition of Complex Biological Mixtures. Application in 1H NMR Metabolomics. Anal Chem, 78, 4281-90.

Dudoit S, Fridlyand J, Speed TP (2002) Comparison of discrimination methods for the classification of tumors using gene expression data. *J Am Stat Assoc*, 97(417), 77-87.

Fisher RA (1936) The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, Part II, 179–88.

Sammon, J. W. (1969) A non-linear mapping for data structure analysis. *IEEE Trans Comput*, C-18 401–409.

Shannon CE (1948) A mathematical theory of communication. *Bell Syst Tech J*, 27(3), 379-423.

van der Maaten, L.J.P. & Hinton, G.E. (2008) Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research*, 9, 2579-2605.

van der Maaten, L.J.P. (2014) Accelerating t-SNE using Tree-Based Algorithms. *Journal of Machine Learning Research*, 15, 3221-3245.