# Package 'TaxicabCA'

September 7, 2026

**Type** Package

**Title** Taxicab Correspondence Analysis

**Version** 0.1.2

**Suggests** GA, testthat

**Imports** grDevices, graphics, stats, utils

**Description** Computation and visualization of Taxicab Correspondence Analysis, Choulakian (2006) <doi:10.1007/s11336-004-1231-4>. Classical correspondence analysis (CA) is a statistical method to analyse 2-dimensional tables of positive numbers and is typically applied to contingency tables (Benzecri, J.-P. (1973). L'Analyse des Donnees. Volume II. L'Analyse des Correspondances. Paris, France: Dunod). Classical CA is based on the Euclidean distance. Taxicab CA is like classical CA but is based on the Taxicab or Manhattan distance. For some tables, Taxicab CA gives more informative results than classical CA.

**License** GPL (>= 2)

**LazyData** true

**RoxygenNote** 7.0.2

**NeedsCompilation** no

**Author** Jacques Allard [aut, cre],
Vartan Choulakian [aut]

**Maintainer** Jacques Allard <jacques.allard@gmail.com>

**Repository** CRAN

**Date/Publication** 2026-09-07 16:50:10 UTC

## Contents

CombineCollinearRowsCols . . . . . 2
ComputeLambda . . . . . 3
CreateAllBinaries . . . . . 4
JitterPosition . . . . . 4
ListToObjects . . . . . 5
milazzese . . . . . 6
plot.tca . . . . . 6

print.tca . . . . . 7
RemoveRowsColumns0sAndDuplicates . . . . . 8
rodent . . . . . 9
saveTCA . . . . . 9
SearchCrissCross . . . . . 10
SearchExhaustive . . . . . 11
SearchGeneticAlgoritm . . . . . 11
summary.tca . . . . . 12
tca . . . . . 13

**Index** **15**

---

`CombineCollinearRowsCols` *Removes rows and columns of zeros and optionnally, row or column duplicates*

---

## Description

Removes rows and columns of zeros and optionnally, row or column duplicates

## Usage

```
CombineCollinearRowsCols(Y, rows = F, cols = F)
```

## Arguments

| | |
|---|---|
| `Y` | A matrix or an object that can be coerced to a matrix |
| `rows` | Logical: Will duplicate rows be removed? |
| `cols` | Logical: Will duplicate columns be removed? |

## Details

Rows and columns of zeros will be removed.

A matrix of zeros will be returned as matrix with 0 row and 0 column.

If rows 1,2,3 are combined, the name of row 1 is kept. Similarly for columns.

## Value

A matrix with rows and columns removed as requested

## Examples

```
CombineCollinearRowsCols(matrix(1:3,nrow=3,ncol=2),cols=TRUE)

CombineCollinearRowsCols(cbind(matrix(1:3,nrow=3,ncol=2),rep(0,3)),cols=TRUE)

CombineCollinearRowsCols(cbind(matrix(1:3,nrow=3,ncol=2),rep(0,3)))

CombineCollinearRowsCols(matrix(0,nrow=3,ncol=3))

CombineCollinearRowsCols(rodent,TRUE,FALSE)
```

---

# ComputeLambda *L1 norm of a projection*

---

## Description

L1 norm of a projection

## Usage

```
ComputeLambda(uFT, pResidual)
```

## Arguments

| | |
|---|---|
| uFT | A vector of 0s and 1s of lenght nc: (-1)^uFT is a unit vector in Linf norm |
| pResidual | A matrix with nc columns |

## Details

This function is for internal usage only.

## Value

L1 norm of the pResidual x (-1)^uFT

## Examples

```
ComputeLambda(uFT=c(FALSE,TRUE,FALSE),pResidual=matrix(1:15,nr=5,nc=3))
```

## CreateAllBinaries *Create all 2^n n-tuples of ab[1] and ab[2]*

### Description

Create all 2^n n-tuples of ab[1] and ab[2]

### Usage

```
CreateAllBinaries(n = 1, ab = c(0, 1))
```

### Arguments

| | |
|---|---|
| n | An integer > 0 |
| ab | A vector of length 2 |

### Details

This function is for internal usage only.

### Value

a n x 2^n matrix

### Examples

```
CreateAllBinaries(3,c(0,1))
```

## JitterPosition *Moves points that are close together*

### Description

Moves points that are close together

### Usage

```
JitterPosition(x, y, jitterDefault = 0.01, Near = 0.045)
```

### Arguments

| | |
|---|---|
| x | The x-coordinates of the points |
| y | The y-coordinates of the points |
| jitterDefault | A positive number controlling the maximum jitter |
| Near | A positive number controlling the definition of "near" |

### Details

This function is for internal usage only.

### Value

A list giving the new x and y coordinates, and, for each point, its number neighbours and a suggested relative size for the plotting symbol

### Examples

```
JitterPosition(c(1:5,2,2,4,4,4),c(1:5,2,2,4,4,4))
```

---

## ListToObjects *Extract objects from a list*

---

### Description

Extract objects from a list to the global environment

### Usage

```
ListToObjects(L, envir = .GlobalEnv)
```

### Arguments

| | |
|---|---|
| L | A list |
| envir | The environment into which variables are created |

### Details

This function is for internal usage only.

### Value

No return

### Examples

```
ListToObjects(list(x=5,A="Hello",M=matrix(1:8,nr=2)),envir=.GlobalEnv)
```

## milazzese *Counts of archeological objects*

### Description

Frequency of object types across the 19 huts of the P. Milazzese settlement in north-eastern Sicily.

### Usage

```
data(milazzese)
```

### Format

A data frame of frequencies with 31 rows and 19 columns

- Columns: Hut
- Rows: Object Type

### Source

Alberti, G., 2013, Making Sense of Contingency Tables in Archaeology: the Aid of Correspondence Analysis to Intra-Site Activity Areas Research, Journal of Data Science 11, 479-499

### Examples

```
tca(milazzese,nAxes=6,algorithm = "criss-cross")
```

## plot.tca *Creates a symmetric plot from a tca-class object*

### Description

Creates a symmetric plot from a tca-class object

### Usage

```
## S3 method for class 'tca'
plot(
  x,
  y = NULL,
  axes = c(1, 2),
  labels.rc = c(0, 1),
  col.rc = c("blue", "red"),
  pch.rc = c(16, 21, 17, 24),
  mass.rc = c(F, F),
  cex.rc = c(NA, NA),
```

```
  jitter = c(T, F),
  ...
)
```

## Arguments

| | |
|---|---|
| x | A tca-class object created by tca |
| y | Unused |
| axes | The two axes to be plotted |
| labels.rc | Two numbers: 0 Symbol only; 1 Label only; 2 Symbol and label |
| col.rc | Colors for rows and columns contributions |
| pch.rc | Plotting characters for rows and columns contributions |
| mass.rc | Logical: Will the area of plotting characters be proportional to mass |
| cex.rc | An overall size factor |
| jitter | Logical: Will close points be moved slightly? |
| ... | Unused. |

## Details

If the number of rows is very large, labels will not be printed.

In this version, jitter is coerced.

## Value

None

## Examples

```
plot(tca(rodent),labels=c(0,1))
```

---

# print.tca *Print result of Taxicab Analysis in easily readable format*

---

## Description

Print result of Taxicab Analysis in easily readable format

## Usage

```
## S3 method for class 'tca'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | A tca tcaObject produced by the function tca |
| ... | Unused |

### Value

An invisible list containing formated outputs

### Examples

```
print(tca(rodent))
```

---

## RemoveRowsColumns0sAndDuplicates

*Removes rows and columns of zeros and optionnally, row or column duplicates*

---

### Description

Removes rows and columns of zeros and optionnally, row or column duplicates

### Usage

```
RemoveRowsColumns0sAndDuplicates(Y, rows = F, cols = F, zeros = F)
```

### Arguments

| | |
|---|---|
| Y | A matrix |
| rows | Logical Will duplicate rows be removed? |
| cols | Logical Will duplicate columns be removed? |
| zeros | Logical Will rows and columns of zeros be removed? |

### Value

A matrix with rows and columns removed as requested

### Examples

```
RemoveRowsColumns0sAndDuplicates(matrix(1:3,nrow=3,ncol=2),cols=TRUE)

RemoveRowsColumns0sAndDuplicates(cbind(matrix(1:3,nrow=3,ncol=2),rep(0,3)),cols=TRUE)

RemoveRowsColumns0sAndDuplicates(cbind(matrix(1:3,nrow=3,ncol=2),rep(0,3)),zeros=TRUE)

RemoveRowsColumns0sAndDuplicates(matrix(0,nrow=3,ncol=3),zeros=TRUE)
```

## rodent *Rodent species abundance*

### Description

Counts of rodents by species and site

### Usage

```
data(rodent)
```

### Format

A data frame of counts with 28 rows and 9 columns

- Columns: Rodent species
- Rows: Location

### Source

Bolger et al. 1997, Response of rodents to habitat fragmentation in coastal Southern California, Ecological Applications 7, 552-563 (as modified and distributed in a University of British Columbia Zoology Department workshop)

### Examples

```
tca(rodent,nAxes=4)
```

## saveTCA *Save tca results to a folder*

### Description

Save tca results to a folder

### Usage

```
saveTCA(
  tcaObject,
  path,
  folder = NULL,
  what = c("report", "csv", "plot", "dataMatrix", "tcaObject"),
  plotAxes = matrix((1:2), nrow = 1, ncol = 2, byrow = T),
  graphicDevice = c("pdf", "postscript", "xfig", "bitmap", "pictex", "cairo_pdf",
    "cairo_ps", "svg", "png", "jpeg", "bmp", "tiff"),
  csvFormat = c("csv", "csv2")
)
```

### Arguments

| | |
|---|---|
| tcaObject | a tca-class object created by tca |
| path | Location of the folder |
| folder | Name of the folder |
| what | What to save: all items specified will be saved |
| plotAxes | A k x 2 matrix giving pairs of axes to plot and save |
| graphicDevice | Format(s) of plots saved. Plots can be saved in more than one format |
| csvFormat | Format of csv files (North American or European) |

### Value

Figure

### Examples

```
saveTCA(tca(rodent),path=tempdir())
```

---

## SearchCrissCross *Search a taxicab principal component using the criss-cross algorithm*

---

### Description

Search a taxicab principal component using the criss-cross algorithm

### Usage

```
SearchCrissCross(pResidual, iterationMax = 20)
```

### Arguments

| | |
|---|---|
| pResidual | A matrix of of non-negative numbers |
| iterationMax | Maximum number of iterations |

### Details

This function is for internal usage only.

The vector uMax is normalize to uMax[1] = 1

### Value

A list: L1Max = maximum L1 norm; uMax = Linf unit vector giving the maximum L1 norm

### Examples

```
SearchCrissCross(matrix(-3:8,nrow=4,ncol=3))
```

## SearchExhaustive *Search a taxicab principal component via exhaustive search*

### Description

Search a taxicab principal component via exhaustive search

### Usage

```
SearchExhaustive(pResidual)
```

### Arguments

| | |
|---|---|
| pResidual | A matrix of of non-negative numbers |

### Details

This function is for internal usage only.

The vector uMax is normalize to uMax[1] = 1

### Value

A list: L1Max = maximum L1 norm; uMax = Linf unit vector giving the maximum L1 norm

### Examples

```
SearchExhaustive(matrix(-3:8,nrow=4,ncol=3))
```

## SearchGeneticAlgoritm *Search a taxicab principal component using the genetic algorithm*

### Description

Search a taxicab principal component using the genetic algorithm

### Usage

```
SearchGeneticAlgoritm(pResidual)
```

### Arguments

| | |
|---|---|
| pResidual | A matrix of of non-negative numbers |

## Details

This function is for internal usage only.

The vector uMax is normalize to uMax[1] = 1

## Value

A list: L1Max = maximum L1 norm; uMax = Linf unit vector giving the maximum L1 norm

## Examples

```
SearchGeneticAlgoritm(matrix(-3:8,nrow=4,ncol=3))
```

---

# summary.tca *Summary of the Taxicab analysis*

---

## Description

Summary of the Taxicab analysis

## Usage

```
## S3 method for class 'tca'
summary(object, ...)
```

## Arguments

| | |
|---|---|
| object | A tca tcaObject produced by the function tca |
| ... | Unused |

## Details

Shows the unstandardized dispersion values

## Value

A list

## Examples

```
summary(tca(rodent))
```

tca *Taxicab Correspondance analysis*

## Description

Computes the Taxicab correspondance analysis of a matrix of non-negative numbers

## Usage

```
tca(
  Y,
  nAxes = 2,
  dataName = NULL,
  combineCollinearRows = c(F, T),
  combineCollinearCols = c(F, T),
  algorithm = c("exhaustive", "criss-cross", "genetic"),
  returnInputMatrix = c(T, F),
  verbose = (nAxes > 2),
  exhaustiveAlgorithmMaxnCol = 20,
  L1MaxDeltaMax = 10^-10
)
```

## Arguments

| | |
|---|---|
| Y | A m x n matrix of non-negative numbers. If Y is not a matrix, the 'as.matrix' transformation will be attempted. Missing values are not allowed. |
| nAxes | Number of axes to compute |
| dataName | A name to be used to identify the outputs in 'plot' and 'saveTCA' () |
| combineCollinearRows | Should collinear rows be combined? |
| combineCollinearCols | Should collinear columns be combined? |
| algorithm | Algorthim requested - may be abreviated to first two letters |
| returnInputMatrix | Will the input matrix be returned |
| verbose | Report progress (default) or not |
| exhaustiveAlgorithmMaxnCol | Maximum size for exhaustive search |
| L1MaxDeltaMax | Change of L1 norm acceptable for convergence in iterative searches |

## Details

Computations are carried out on the transposed matrix if nrow(Y) < ncol(Y). In the following, we assume that nrow(Y) >= ncol(Y)

Row and column names will be created if necessary.

Zeros rows and columns are removed.

If ncol(Y) <= exhaustiveAlgorithmMaxnCol the exhaustive algorithm used unless otherwise specified.

If ncol(Y) > exhaustiveAlgorithmMaxnCol the genetic algorithm used unless otherwise specified.

Algorithm = exhaustive is overridden if ncol(Y) > exhaustiveAlgorithmMaxnCol.

For ncol(Y) <= exhaustiveAlgorithmMaxnCol, the user may want to specify algorithm = genetic is nrow(Y) is very large, since exhaustive computation may be slow.

If ncol(Y) <= exhaustiveAlgorithmMaxnCol the genetic algorithm is used unless otherwise specified.

(ncol(Y) = 20 appears to be the maximum practical on 2017 vintage Intel-based desktops).

## Value

A list with class 'tca' containing the following components:

| | |
|---|---|
| `dispersion` | A nAxes-length vector of matrix of column contributions |
| `rowScores` | A m x nAxes matrix of column contributions |
| `colScores` | A nAxes x n matrix of row contributions |
| `rowMass` | Row weights: apply(Y,1,sum)/sum(Y) |
| `colMass` | Column weights: apply(Y,2,sum)/sum(Y) |
| `dataName` | A name to be used to identify the output in 'plot' and 'save' |
| `algorithm` | Algorithm used (may be different from the algorythm requested) |
| `dataMatrixTotal` | Sum of the input matrix entries |
| `dataMatrix` | The matrix used in the computation |
| `rowColCombined` | A list describing removed or combined rows and columns, if any |

## Examples

```
tca(rodent,nAxes=4)
tca(rodent,nAxes=4,combineCollinearRows=c(TRUE,FALSE))
```

# Index

∗ **datasets**
    milazzese, 6
    rodent, 9

CombineCollinearRowsCols, 2
ComputeLambda, 3
CreateAllBinaries, 4

JitterPosition, 4

ListToObjects, 5

milazzese, 6

plot.tca, 6
print.tca, 7

RemoveRowsColumns0sAndDuplicates, 8
rodent, 9

saveTCA, 9
SearchCrissCross, 10
SearchExhaustive, 11
SearchGeneticAlgoritm, 11
summary.tca, 12

tca, 13