Package 'sequence'

February 11, 2020

Version 2.0

Date 2020-01-11

Title Analysis of Sequences of Events

Author Jean-Sebastien Pierre <`jean-sebastien.pierre@univ-rennes1.fr`>

Maintainer Jean-Sebastien Pierre <`jean-sebastien.pierre@univ-rennes1.fr`>

Depends R (>= 3.5.0)

Description Pierre, J. S. and Kasper, C.,(1990), ``The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship." Biology of Behaviour 15(3-4): 125-151.(in french). Includes the calculation of transition matrices, their manipulation (symmetrization and diagonal loading), their analysis by CA (Correspondence Analysis), their comparison by likelihood ratio tests, the graphic plot of flow charts on the factorial axes of the CA.

URL none

NeedsCompilation no

Repository CRAN

Date/Publication 2020-02-11 10:00:02 UTC

R topics documented:

sequence-package 2
aphelinus 3
aphmat 4
ca 5
compmat 6
compseq 8
courtil 9
flux 11
geneseq 13
homoth 14
makedico 15
makeSeries 16

matrcom 17
mattrans 18
multMatrans 19
read.old.seq 20
read.series.seq 21
rotation 22
seriseq 23
seriseq2 24
smartArrow 25
symet 27
wiepkema 28
xcsa 29

Index **31**

`sequence-package` *Analysis of behavioral (or of any other type) sequences.*

Description

Builds transition/succession matrices (lag analysis), compares several matrices. Plane drawing of flow charts obtained from those matrices by a method based on a modified Correspondence Analysis (CA). The modification consists in symmetrising the matrix and loading its diagonal following Foucart (1985) and Van der Heijden (1986). This leads to a 'nice' positionning of the edges of the graph, representing the different items in succession. 'Smart' arrows are drawn as vertices.

Details

`read.series.seq` reads a series of sequences, each of them considered as an identifiers list. `compseq` compares the sequences by likelihood ratio tests (LRT) with a Bonferroni correction. The LRT serve to build a distance matrix of the sequences two-by-two. This matrix of distance is used both for a clustering of them and for a mapping by Principal Coordinates Analysis (PCA) also called Multidimensional Scaling (MDS).

From the sequences a transition matrix is obtained. `symet` symmetrises and/or loads its diagonal. `ca` makes a Correspondence Analysis and saves the factorial coordinates as an object. `flux` draws the transition graph. The complete analysis chain is the following:
read.series.seq -> compseq -> symet -> ca -> flux
A sketch of the method is given in Pierre and Kasper (1990).

Author(s)

Jean-Sebastien Pierre `<Jean-sebastien.pierre@univ-rennes1.fr>`

References

Pierre, J. S. and C. Kasper (1990). "The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship." Biology of Behaviour 15(3-4): 125-151.
Foucart, T. (1985). "Tableaux symetriques et tableaux d'echanges." Revue statistiques appliquee 33: 37-53.
Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

`aphelinus` *Flight behaviour of* Aphelinus abdominalis

Description

Flight behaviour of the parasitic hymenoptera *Aphelinus abdominalis* in flight tunnel.

Usage

```
data(aphelinus)
```

Format

The format is: List of 10 sequences on the following pattern:
aphelinus[[1]] =list("%1","B","TD","TH")
and so on until aphelinus[[10]]

Details

A sequences of 10 individuals

Source

Anne Le Ralec, personnal communication.

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.(in French)

Examples

```
data(aphelinus)
u=compseq(aphelinus)
mat=u$mcom
print(mat)
```

`aphmat` *Transition matrices of the flight behaviour of 10 individuals of* Aphelinus abdominalis

---

Description

10 succession matrices extracted from the object `aphelinus`. Matrices are provided as `data.frames`

Usage

```
data(aphmat)
```

Format

The format is: List of 10 `data.frames`
aphmat[[1]] :'data.frame': 5 obs. of 5 variables:
B : num [1:5] 0 0 0 0 0
E0: num [1:5] 0 0 0 0 0
TD: num [1:5] 1 0 0 0 0
TH: num [1:5] 0 0 1 0 0
V: num [1:5] 0 0 0 0 0
and so on until aphmat[[10]]

Source

Personal communication by courtesy of Anne Le Ralec

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.(in French)

Examples

```
data(aphmat)
compmat(aphmat)
```

`ca` *Correspondence analysis*

Description

Achieves a Correspondence Analysis (CA) on a numeric table of class data.frame

Usage

```
ca(x, nfac = 3, isup = 0, jsup = 0, histev = FALSE, grr = FALSE, grc = FALSE,
grrc = FALSE, grlist = rbind(c(1, 2), c(1, 3), c(2, 3)), prtm = FALSE,
prtevr = FALSE, prtevc = FALSE, eps = 1e-09)
```

Arguments

`x`	data.frame minimal dimension 4 x 3. The first column **must contain the character strings of the identifiers of raws** any other type, class or dimension results in an error and in the program break.
`nfac`	Number of factors to retain (maximum 7)
`isup`	list of illustrative rows. 0 = no illustrative rows (default)
`jsup`	List of illustrative columns. Same as isup.
`histev`	Boolean : whether to plot or not the histogram of eigenvalues.
`grr`	Boolean : plot the graph of rows on the axes defined by grlist.
`grc`	Boolean : Plot the graph of columns on the axes defined by grlist.
`grrc`	Boolean : Plot the simultaneous graph of rows and columns on the axes defined by grlist. Labels of rows in black, labels of columns in red.
`grlist`	matrix: defines the factorial plans to plot. See details for an example.
`prtm`	Boolean: Print or not the data frame. Default = FALSE
`prtevr`	Boolean: Print or not the rows eigenvectors. Default = FALSE
`prtevc`	Boolean: Print or not the columns eigenvectors. Default = FALSE
`eps`	numeric: (tolerance) Precision for null eigenvalues. Default = 10E-09

Details

grlist: the successive plots to draw are defined by a matrix of dimension k,2. k = number of plans to plot. Example: to plot the plans 1-2, 1-3 and 2-3 enter sometning as matrix(1,2,1,3,2,3,nrow=3,ncol=2,byrow=2) or rbind(c(1,2),c(1,3),c(2,3)). **Markovian matrix:** In the case of a Markovian or of a transition matrix, one can symetrise (X + t(X)) and load it (sum of the margins added to the diagonal, before applying CA (cf `See Also`).
In the case of a markovian square matrix (succession or transition matrix) one can symmetrize and load it (`symet`) before representing it by a graph (`flux`)

Value

An object of class ca with attributes

`fr`	data.frame: weight and factorial coordinates of each row (principal and illustrative). The attribute `type` has the value "pri" for principal and "ill" for illustrative
`fc`	data.frame: weight and factorial coordinates of each column (principal and illustrative). `type` as in fr

Author(s)

Jean-Sebastien Pierre `<Jean-sebastien.pierre@univ-rennes1.fr>`

References

Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

See Also

`princomp`, `compseq` to build a transition matrix,
`symet` to modify it (symmetrization and diagonal loading), `flux` for the design of a graph.

Examples

```
# On Csa data (xcsa)
library(sequence)
   data(xcsa)
   ca(xcsa)
```

`compmat` *Comparison of transition (or succession) matrices*

Description

Compares statistically succession matrices by likelihood ratio tests. Performs also a cluster analysis of the sequences and a Pricipal Coordinates Analysis (PCA) on the distance matrix between them.

Usage

```
compmat(serMat, alpha = 0.05, meth = "ward.D", printdata = FALSE, printdico = FALSE,
printmat = FALSE, eps = 1e-07, clust = TRUE, pca = TRUE)
```

Arguments

`serMat`	List of data.frames. Each of them must contain a matrix of identical dimension, with the same row.names.
`alpha`	numeric: global risk threshold for pairwise comparisons. Default = 0.05
`meth`	character: Clustering method. cf `hclust`.
`printdata`	Boolean: Print original list of matrices.
`printdico`	Boolean:Print the dictionnary of states from `ser`.
`printmat`	Boolean: print all transition matrices and the consensus matrix.
`eps`	numeric: precision for the convergence of `cmdscale`.
`clust`	Boolean: performs cluster analysis.
`pca`	Boolean: performs a Principal Coordinates Analysis.

Details

The log likelihood ratio times -2 is used both for tests (Chi-Square approximation followed by Bonferroni post hoc tests) and as a distance to cluster the sequences and to represent them on factorial plans (Principal Coordinates Analysis). Warning: not a metric distance. Susceptible to give incoherent clustering with some methods (`meth`).

Value

an object of class compseq with attributes

- dico Dictionnary of states
- mdist Matrix of pairwise distances between sequences
- msign Matrix of pairwise significance levels between sequences
- mcom Common or consensus transition matrix

Author(s)

Jean-Sebastien Pierre `<Jean-sebastien.pierre@univ-rennes1.fr>`

References

a Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.
Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

See Also

`hclust`, `cmdscale`, `ca`

Examples

```
# Compares 10 transition matrices in \code{aphmat}
data(aphmat)
compmat(aphmat,clust=FALSE,pca=FALSE)
```

`compseq` *Comparison of behavioral (or any) sequences*

Description

Compares statistically sequences of states (behavior, texts, molecular data) by likelihood ratio tests on their markovian transition matrices. Performs also a cluster analysis of the sequences and a Principal Coordinates Analysis on the distance matrix between them.

Usage

```
compseq(ser,alpha=0.05,meth="ward.D",printdata=FALSE,printdico=TRUE,printmat=FALSE,
eps=1e-07,clust=TRUE,pca=TRUE)
```

Arguments

`ser`	list of list: set of sequences
`alpha`	numeric: global risk threshold for pairwise comparisons.
`meth`	character:Clustering method. cf `hclust`.
`printdata`	Boolean:Print original data.
`printdico`	Boolean:Print the dictionnary of states from `ser`.
`printmat`	Boolean: print all transition matrices and the consensus matrix.
`eps`	numeric: precision for the convergence of `cmdscale`.
`clust`	Boolean: do the cluster analysis.
`pca`	Boolean: do the principal coordinates analysis.

Details

The log likelihood ratio times -2 is used both for tests (Chi-Square approximation followed by Bonferroni post hoc tests) and as a distance to cluster the sequences and to represent them on factorial plans (Principal Coordinates Analysis). Warning: not a metric distance. Susceptible to give incoherent clustering with some methods (`meth`).
This function does essentially the same work as `compmat` but with matrices instead of sequences entry.

Value

an object of class compseq with attributes

- dico Dictionnary of states
- mdist Matrix of pairwise distances between sequences
- msign Matrix of pairwise significance levels between sequences
- mcom Common or consensus transition matrix

Author(s)

Jean-Sebastien Pierre
`<Jean-sebastien.pierre@univ-rennes1.fr>`

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.
Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

See Also

`hclust`, `cmdscale`, `ca`, `flux`

Examples

```
data(seriseq)
compseq(seriseq)
```

`courtil` *Sexual courtship of the male molecricket*

Description

Succession matrix of the patterns of male molecricket (*Gryllotalpa gryllotalpa* orthoptera)

Usage

```
data(courtil)
```

Format

A data frame with 28 observations on the following 29 variables.

`id` a factor with levels
`AA` `AA+` `AAB` `ACC` `ACQ` `BAL` `CH` `CP1` `DIV` `E` `FR` `FUI` `IMM` `INV` `LA` `M` `MB` `P1` `PLP` `PRJ` `R` `RET` `RV` `SP1` `TR` `VAV` `VC` `VCH`

`DIV` diverse (walking, feeding, grooming)

`VAV` go to

`AA.` lowers wings

`ACQ` antenna-cerques contact

`P1` rise P1

`CP1` kicking with forelegs (P1)

`SP1` joins and rises forelegs

`LA` rises antennae and rehects them backward

`PRJ` projection of liquid from anus

`RET` returns

`FUI` escapes

`IMM` freezes

`R` steps back

`BAL` rocking

`FR` wings rustling

`TR` short and acute sound ("treets")

`CH` call song

`PLP` palpation of the female

`E` avoids

`INV` turns around and approaches

`AA` lowers wings at maximum

`VCH` vibers and sings

`AAB` lowers wings and abdomen

`VC` vibration of cerques

`RV` steps back lowering wings and abdomen

`M` inserts backward under the female which climbs on the mail

`ACC` mating

`MB` 'masturbation' - rubbing of genitalia with p3 tarsa

Details

Matrix of succession of patterns, provided as a `data.frame`. The code of the rows, character values of the levels of the factor `id` are the same as the names of the columns after `id`. The elements of the matrix. The value of each element reflects the number of times the pattern `id[i]` is followed by the pattern names`(courtil)[j+1]`.

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.(in french)
Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

Examples

```
data(courtil)
symet(courtil,ident=TRUE)->scourtil
ca(scourtil,grr=TRUE)
```

`flux` *Draws a flow chart from a transition matrix*

Description

Draws a flow chart from a transition matrix on the projection plans of a CA (Correspondence Analysis). requires two objects of class `data.frame`: `tabcoord` a table of axis coordinates on up to 7 columns, and `tabtr` the transition matrix.

Usage

```
flux(tabcoord, tabtr, dic = NULL, fac = c(1, 2), Sort = NULL, threshold = 0,
scale = 0.1, cscale = 0.1,main="Transition graph", ...)
```

Arguments

`tabcoord`	data.frame: nfac + 3 columns • id factor: identifier of individuals • w numeric: weight • f1 numeric: coordinates on axis 1 • f2,...numeric: next coordinates until nfact • type factor: "pri" for principal, "ill" for illustrative (or 'supplementary')
`tabtr`	data.frame: transition matrix. Either frequencies or probabilities. **Caution** this data.frame must contain the row identifiers as first column
`dic`	character vector: list of items labels. If NULL (default) must be extractible from `tabtr` column names
`fac`	vector: two elements: the numbers of the two factors to use in `tabcoord`
`Sort`	numeric: number of the factor used to sort the drawing of the elements. Default = NULL (sorting in the order of the dictionary. Otherwise must be comprised between 1 et nfac. If positive, sort in ascending order. If negative, sort in decreasing order

`threshold`	numeric: minimal threshold to draw the arrows. Represent a oroportion of the weight of the element used as origin of the arrow.
`scale`	numeric: Controls the width of the arrows
`cscale`	numeric: Controls the radius of the circles.
`main`	character:title of the plot
`...`	... supplementary arguments for `smartArrow`

Details

`tabcoord` must be an the attribute $fl or $fc of an object of class `ca`

Value

An object of class `graph`.

Note

The correct definition of fac is under the user's responsability. **Must know** how many coordinates exist in the object tabcoord. Use `summary(<tabcoord>)` in case of doubt.

Author(s)

Jean-Sebastien Pierre
`<Jean-sebastien.pierre@univ-rennes1.fr>`

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.(in french)
Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

See Also

`compseq` to create transition matrices, `compmat` to compare transition matrices,
`read.series.seq` to read a file of sequences,`ca` to perform a Correspondence Analysis

Examples

```
# Analysis of Wiepkema's data
data(wiepkema)
# Symmetrisation and diagonal loading
symet(wiepkema,ident=TRUE)->swiepkema
# Correspondence Analysis
ca(wiepkema,isup=c(5,11),jsup=c(5,11))->afcwiepk
# Flow chart / transition graph
```

```
flux(tabcoord=afcwiepk$fr,tabtr=wiepkema,threshold=0.025,scale=2)
```

`geneseq` *Generation aof random sequences*

Description

Generates `nseq` random sequences of items represented by codes, either by random sampling (order 0), either using a Discrete Time Markov Chain (DTMC) transition matrix (order 1).

Usage

```
geneseq(nseq, lmin, lmax, order = 0, dico = NULL, mattrans = NULL)
```

Arguments

`nseq`	numeric: number of sequences to generate.
`lmin`	numeric: minimum length of sequences
`lmax`	numeric: maximum length of sequences
`order`	numeric: order of the Markov Process, 0 or 1. See details.
`dico`	character vector: dictionnary of items.
`mattrans`	data.frame: transition matrix with identifiers as first column.

Details

Generates random sequences of states according to a Markov process of order 0 or 1.
order 0: the following state is drawn with equal probabilities in the item list, no influence of previous state.
order 1: the following state is drawn with a probability proportional to the product of the transition matrix by the vector of states.
`nseq` sequences of random size are generated. The size of each sequence is a random integer drawn under uniform probability between `lmin` and `lmax` (both included).

Value

A list of nseq items

`A1`	list: first sequence
`A2`	list: second sequence

... and so on until `nseq`.

Note

The transition matrix must be square and Markovian. A non-square matrix produces an error and program break. Its columns must all sum to 1. No verification is done for this last point. A matrix in which the columns do not sum to 1 may lead to spurious or unpredictible results.

Author(s)

Jean-Sebastien Pierre
`<Jean-sebastien.pierre@univ-rennes1.fr>`

See Also

`compseq`

Examples

```
# 1 - generates 5 random sequences of length from 1 to 15 with
# symbols a,b,c,d.(order = 0)
geneseq(5,1,15,dico=letters[1:4])
mat<-data.frame(id=letters[1:3],a=c(0,0.75,0.25),b=c(0.5,0.25,0.25),
c=c(0.1,0.8,0.1))
# 2 - generates 2 random sequences of length 10 with symbols a,b,c,d
# according to the transition matrix mat
geneseq(2,10,10,order=1,mattrans=mat)
```

`homoth` *Homothety*

Description

Homothety (scaling) of center `centre` and of factor `scale` on an object `form`

Usage

```
homoth(form = rbind(c(0, 1), c(1, 1)), centre = c(0, 1), scale = 0.5)
```

Arguments

`form`	matrix: with two columns x and y. Set of points making the shape
`centre`	vector: Coordinates of the homothety center
`scale`	numeric: Homothety ratio

Details

Utilitary used by `smartArrow`

Value

A matrix, transformed from `form` after homothety.

Author(s)

Jean-Sebastien Pierre
`<Jean-sebastien.pierre@univ-rennes1.fr>`

See Also

`smartArrow`

Examples

```
homoth(rbind(c(0,sqrt(2)/2),c(0,sqrt(2)/2)),c(0,0),2)
```

`makedico` *Build the dictionary of the sequences*

Description

Assesses the dictionnary common to various sequences, counts the occurences of each word (item, state)

Usage

```
makedico(x, printdata = FALSE, printdico = FALSE)
```

Arguments

`x`	sequence: an object of class sequence
`printdata`	`printdata` logical: print sequences or not.
`printdico`	`printdico` logical: print or not the resulting dictionnary.

Value

A list of character strings

Author(s)

Jean-Sebastien Pierre email: Jean-sebastien.pierre@univ-rennes1.fr

Examples

```
data(seriseq)
dic=makedico(seriseq,printdico=TRUE)
```

`makeSeries` *Builds a list of list from a vector and a factor*

Description

Builds an object of class list of lists, from two columns, of a `data.frame`. The first one contains the succession of codes to use, the second is a factor adequate to split the series into separate bouts.

Usage

```
makeSeries(x, fac)
```

Arguments

x	`character` `vector`: the record of item codes under study
fac	`factor`: a factor, of same length as x, indicating how to split the sequences into separate bouts.

Details

Suppose we observe the succession a,c,d,a,a,f on the subject A and b,d,f,e,c,c,a,d on the subject B. This can be coded as the concatenation
x: a,c,d,a,a,f,b,d,f,e,c,c,a,d
fac: A,A,A,A,A,A,B,B,B,B,B,B,B,B

fac indicating that the 6 first itms were successively emitted by A and the 8 following by B. See example.

Value

a list of lists, one per level of the factor `fac`

Author(s)

Jean-Sebastien Pierre

See Also

`read.old.seq`, `read.series.seq`,`compseq`

Examples

```
ser=sample(letters[1:6],100,replace=TRUE)
fact=as.factor(c(rep("one",30),rep("two",20),rep("three",10),rep("four",40)))
print(rbind(ser,fact)) # Facultative to see the structure
seq=makeSeries(ser,fact)
print(seq)
compseq(seq) # Facultative, compares the sequences obtained.
```

`matrcom` *Calculates a common succession matrix*

Description

Computes the succession matrix common to a set of sequences.

Usage

```
matrcom(x, printdata = FALSE, printdico = FALSE, printmat = FALSE, printcom = TRUE)
```

Arguments

`x`	seq: list of lists of class `character`. The sequences to analyse.
`printdata`	logical: print all sequences.
`printdico`	logical: print the dictionary.
`printmat`	logical: print each individual matrix.
`printcom`	logical: print the common succession matrix.

Details

The resulting matrix (succession: matrix) gives the number of times the element i (row) is followed by the element j (column).

Value

data.frame: the succession matrix with the label of items as first column.

Author(s)

Jean-Sebastien Pierre `<Jean-sebastien.pierre@univ-rennes1.fr>`

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.(in french)
Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

See Also

`flux`, `ca`

Examples

```
data(seriseq)
matrcom(seriseq)
```

`mattrans` *Creates of a succession matrix*

Description

Creates a succession matrix of order 1 from an object of class `sequence`.

Usage

```
mattrans(x, dico = NULL, print = FALSE)
```

Arguments

`x`	Cseq: sequence
`dico`	character: list of items codes
`print`	logical: print the matrix. Default = FALSE

Details

Creates a succession, not a transition matrix. Integer numbers (frequencies) and not conditional probabilities. Function used by `compseq`

Value

a matrix: the succession matrix.

Note

Not supposed to be used alone; if needed, the user is invited to use rather `matrcom`

Author(s)

Jean-Sebastien Pierre <`Jean-sebastien.pierre@univ-rennes1.fr`>

See Also

`compseq` ,`matrcom`

Examples

```
data(seriseq)
mattrans(unlist(seriseq))
```

`multMatrans` *Converts a list of sequences into a list of transition matrices*

Description

Converts a list of sequences into a list of transition matrices. Matrices are square, of same size, and provided as `data.frame`.

Usage

```
multMatrans(ser, printdata = FALSE, printdico = TRUE, printmat = FALSE)
```

Arguments

`ser`	The list of sequences
`printdata`	: print the input sequences
`printdico`	: Prints the dictionary of items
`printmat`	: Print the list of transition matrices

Value

An object of class `lmat`

`dico`	: dictionnary of sequences components
`listmat`	The list of data.frames

Author(s)

Jean-Sebastien Pierre
`<Jean-sebastien.pierre@univ-rennes1.fr>`

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.(in french)

Examples

```
data(aphelinus)
aphMat=multMatrans(aphelinus)
aphMat$listmat
```

`read.old.seq` *Function to read the deprecated files *.SEQ*

Description

Used to recover the sequence files used in a former DOS version of the program.

Usage

```
read.old.seq(file = "NULL")
```

Arguments

`file`	Fichier texte a lire

Details

Only used for compatibility with the old format *.SEQ of program `graflux`: Items coded by at most 4 characters, One code per line. The separator of sequences is a code of 4 characters beginning by %.
Example:

%001
AAAA
BBBB
EEEE
DDDD
AAAA
FFFF
CCCC
FFFF
%002
FFFF
BBBB
CCCC
....

Value

An object de classe `sequence`, i.e. a list of lists.

Author(s)

Jean-Sebastien Pierre `<Jean-sebastien.pierre\@univ-rennes1.fr>`

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the Male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.(in french)
Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

See Also

`read.series.seq` for a more convenient format, `compseq` for the analysis

Examples

```
sink("old-seq")
cat("%001\n")
cat("aaa\n")
cat("bbb\n")
cat("aaa\n")
cat("ccc\n")
cat("%002\n")
cat("ccc\n")
cat("ccc\n")
cat("aaa\n")
cat("aaa\n")
cat("bbb\n")
sink()
seq1<-read.old.seq("old-seq")
seq1
file.remove("old-seq")
```

`read.series.seq` *Reads a series of sequences.*

Description

Input from a text file of a series of behavioral (or any other type) sequences. Builds an object of class `sequence`

Usage

```
read.series.seq(fich = NULL)
```

Arguments

`fich`	character: name of the text file to read

Details

The text file must have one line by sequence. Each line must begin by a character identifier of the sequence, followed by a tab character or a space, followed by the series of item codes, separated also by tabs or spaces. example:

A1 A B A C D A B C
A2 A D D C A B
A3 B D C A D A B C
A4 C B C A
A5 C D A B C

Value

An object of class `sequence`

Author(s)

Jean-Sebastien Pierre
`<Jean-sebastien.pierre@univ-rennes1.fr>`

See Also

`compseq`

Examples

```
sink("series-seq")
cat("A1 A B A C D A B C\n")
cat("A2 A D D C A B\n")
cat("A3 B D C A D A B C\n")
cat("A4 C B C A\n")
cat("A5 C D A B C\n")
sink()
read.series.seq("series-seq")
file.remove("series-seq")
```

`rotation` *Plane rotation*

Description

Rotation of the object `form` around a center `center`, of angle `angle`. Angle in radians in the direct (trigonometric) direction.

Usage

```
rotation(form, center, angle)
```

Arguments

`form`	matrix: two columns, x and y, giving the coordinates of the points composing the object.
`center`	vector: coordinates of the rotation center.
`angle`	numeric: rotation angle in radians

Details

A geometric function used by `smartArrow`

Value

matrix: `form` object rotated.

Author(s)

Jean-Sebastien Pierre `<Jean-sebastien.pierre@univ-rennes1.fr>`

See Also

`smartArrow`

Examples

```
rotation(cbind(c(0,1),c(0,0)),c(0,0),pi/3)
```

`seriseq` *Random series*

Description

5 random series made of letters "a" to "f"

Usage

```
data(seriseq)
```

Format

The format is: List of 5 lists

[[1]]	chr	[1:9]	"A1"	"d"	"e"	"a"	...
[[2]]	chr	[1:8]	"A2"	"f"	"d"	"b"	...
[[3]]	chr	[1:9]	"A3"	"c"	"e"	"b"	...
[[4]]	chr	[1:8]	"A4"	"e"	"d"	"b"	...
[[5]]	chr	[1:8]	"A5"	"d"	"b"	"f"	...

Details

Example of format used by `compseq` and built from a text file by the function `read.series.seq`. List of lists. Each elementary list represents a particular sequence which must be compared to all others.

Author(s)

Jean-Sebastien Pierre `<Jean-sebastien.pierre@univ-rennes1.fr>`

Source

Random simulation made by J.S. Pierre.

See Also

`read.series.seq`

Examples

```
data(seriseq)
compseq(seriseq)
```

`seriseq2` *random series*

Description

25 random series made of letters "a" to "f".

Usage

```
data(seriseq2)
```

Format

List of 25 lists

[[1]]	chr	[1:23]	"A1"	"d"	"d"	"d"	...
[[2]]	chr	[1:31]	"A2"	"b"	"b"	"c"	...
[[3]]	chr	[1:27]	"A3"	"f"	"e"	"e"	...
[[4]]	chr	[1:12]	"A4"	"a"	"c"	"a"	...
[[5]]	chr	[1:9]	"A5"	"a"	"b"	"e"	...
[[6]]	chr	[1:11]	"A6"	"e"	"f"	"a"	...
[[7]]	chr	[1:30]	"A7"	"f"	"d"	"f"	...
[[8]]	chr	[1:29]	"A8"	"b"	"a"	"e"	...
[[9]]	chr	[1:24]	"A9"	"d"	"b"	"a"	...
[[10]]	chr	[1:12]	"A10"	"a"	"f"	"b"	...
[[11]]	chr	[1:18]	"A11"	"d"	"d"	"e"	...

[[12]]	chr	[1:22]	"A12"	"e"	"c"	"e"	...
[[13]]	chr	[1:25]	"A13"	"e"	"f"	"f"	...
[[14]]	chr	[1:20]	"A14"	"c"	"f"	"a"	...
[[15]]	chr	[1:29]	"A15"	"e"	"c"	"c"	...
[[16]]	chr	[1:19]	"A16"	"c"	"a"	"f"	...
[[17]]	chr	[1:25]	"A17"	"f"	"e"	"d"	...
[[18]]	chr	[1:18]	"A18"	"d"	"c"	"e"	...
[[19]]	chr	[1:25]	"A19"	"c"	"a"	"b"	...
[[20]]	chr	[1:8]	"A20"	"f"	"d"	"a"	...
[[21]]	chr	[1:14]	"A21"	"c"	"e"	"e"	...
[[22]]	chr	[1:13]	"A22"	"f"	"c"	"e"	...
[[23]]	chr	[1:28]	"A23"	"b"	"e"	"d"	...
[[24]]	chr	[1:18]	"A24"	"e"	"f"	"a"	...
[[25]]	chr	[1:29]	"A25"	"e"	"a"	"b"	...

Details

Example of format used by `compseq` and built from a text file by the function `read.serie.seq`. List of lists. Each elementary list represents a particular sequence which must be compared to all others.

Source

Random simulation made by J.S. Pierre

Examples

```
data(seriseq2)
compseq(seriseq2)
```

`smartArrow` *Draws a ’smart’ arrow between two points.*

Description

Draws a ’smart’ arrow between two points surrounded by a circle of known radius. The arrow is arc shaped and ends with an arrowhead.

Usage

```
smartArrow(A = c(0, 2), B = c(2, 2), Ra = 0.2, Rb = 0.1, ClegA = "A",
ClegB = "B", width = 0.1, col = "lightgreen", ccol = "yellow",
density = NULL, angle = 0, marge = 1.3, debord = 1.4, plot = FALSE,
trace = TRUE)
```

Arguments

`A`	numeric vector: coordinates of point A.
`B`	numeric vector: coordinatess of point B.
`Ra`	numeric: Radius of circle A
`Rb`	numeric: Radius of circle B
`ClegA`	character: label of circle A
`ClegB`	character: label of circle B
`width`	numeric: width of the arrow
`col`	color to fill the arrow. See`polygon`
`ccol`	color to fill the circle. See `symbols`
`density`	numeric: density of shading lines per inch (default=NULL). See `polygon`
`angle`	numeric: angle of shading lines in radians (default = 0). See `polygon`
`marge`	numeric: Angular distance between each circle and the corresponding end of the arrow (default = 1.3).
`debord`	numeric: overflow of the arrowhead relatively to its body. (Default = 1.4)
`plot`	logical: to draw a first arrow on a new plot.(Default = FALSE)
`trace`	logical: if TRUE (default) the arrow is drawn, else it is simply calculated.

Details

SmartArrow can work both as a high level and a low level function, depending on the value of the argument `plot`.
Used in `flux`. Not supposed to be used directly by the user.

Value

An object of class `smartarrow` having attributes:

`arrow`	The polygon representing the arrow.
`bg`	Color of background
`A`	Coordinates of point A
`B`	Coordinates of point B
`Ra`	Radius of circle A
`Rb`	Radius of circle B
`left`	The most extreme left point of the set circles-arrows.
`right`	The most extreme right point of this set
`up`	The most extreme high point.
`down`	The most extreme low point
`drawcircles`	boolean : draw the circles or not.

Author(s)

Jean-Sebastien Pierre
`<Jean-sebastien.pierre@univ-rennes1.fr>`

See Also

`flux`, `polygon`,`symbols`
`arrows` ~ for simple arrows

Examples

```
# For a quick illustration
   smartArrow(plot=TRUE)
# an arrow from a randomly drawn point on [0,2]x[0,2]
   smartArrow(B=runif(2,0,2),Rb=runif(1,0,0.5),ClegB="C")
```

`symet` *Symmetrization and diagonal loading of a transition matrix*

Description

Adds the matrix to its transpose (X + t(X)) and loads its diagonal by the sum of each row.

Usage

```
symet(x, sym = TRUE, charge = TRUE, ident = FALSE)
```

Arguments

`x`	data.frame: The matrix to process as a data.frame
`sym`	logical: symmetrisation.
`charge`	logical: diagonal loading.
`ident`	logical: first column consists in identifiers.

Value

a data.frame: the resulting matrix

Author(s)

Jean-Sebastien Pierre
`<Jean-sebastien.pierre@univ-rennes1.fr>`

References

Pierre, J. S. and C. Kasper (1990). The Design of Ethological Flow-Charts on Factorial Analysis Representations - an Application to the Study of the male Mole-Cricket Sexual Courtship. Biology of Behaviour 15(3-4): 125-151.
Van der Heijden, P. G. M. 1986. Transition matrices, model fitting and correspondence analysis. In: Data Analysis and Informatics IV (Ed. by E. Diday), pp. 221-226. Elsevier Science Publishers.

See Also

`flux`,`matrcom`

Examples

```
data(wiepkema)
swiepkema<-symet(wiepkema,ident=TRUE)
print(swiepkema)
```

`wiepkema` *Wiepkema's (1961) historical data*

Description

Succession matrix for the sexual courtship of the gorchak fish (*Rhodeus sericeus*, cyprinidae)

Usage

```
data(wiepkema)
```

Format

A data frame with 12 observations on the following 13 variables.

`id` factor: levels `chf` `chs` `ffl` `fl` `hb` `hdp` `jk` `le` `qu` `sk` `sn` `tu`

`jk` numeric: jerking

`tu` numeric: turning

`hb` numeric: head butting

`chs` numeric: chasing

`fl` numeric: fleeing

`qu` numeric: quivering

`le` numeric: leading

`hdp` numeric: head-down

`sk` numerics: kimming

`sn` numeric: snapping

`chf` numeric: chafing

`ffl` numeric: finflickering

Details

Historical data of Wipkema (1961) consisting in the succession of patterns in the sexual courtship of a fish. Treated by this author by Principal Component Analysis.

Source

Wiepkema, P. R. 1961. An ethological analysis of the reproductive behaviour of the bitterling. Arch Neerl Zool, 14, 103-199.

References

Van der Heijden, P. G. M. (1990). Correspondance analysis of transition matrices, with special attention to missing entries and asymmetry. Animal Behaviour, 39, 49-64.

Examples

```
data(wiepkema)
# Symmetrization and diagonal loading before a CA
swiepkema <- symet(wiepkema,ident=TRUE)
ca(swiepkema)
```

`xcsa` *French CSA (1976) data*

Description

Medical attendence depending on socio-professional categories.

Usage

```
data(xcsa)
```

Format

A data frame with 15 observations on the following 7 variables.

- `id` factor: Socio-professional categories: levels: `acm` `fcm` `fco` `fem` `foo` `foq` `fpl` `fse` `hco` `hem` `hma` `hoq` `hos` `hpl` `hse` `sup` `hcm` `fos`
- `ge1` visited a general practitioner during the previous year
- `ge0` did not visit a general practitioner during the previous year
- `sp1` visited a specialist during the previous year
- `sp0` did not visite a specialist during the previous year
- `de1` visited a dentist during the previous year
- `de0` did not visite a dentist during the previous year

Details

Detail of socio-professional categories codes :
First letter = h : man
First letter = f : woman
cm : middle manager
co : manager, unprecised status
pl : Senior executive or liberal profession
oq : qualified worker
se : unemployed
os : specialised worker
ma : unskilled worker

Source

Ronald Cehessat (1976) exercices de statistique et informatique appliquees Dunod (Paris)

Examples

```
data(xcsa)
ca(xcsa)
```

Index

∗Topic **datasets**
  aphelinus, 3
  aphmat, 4
  courtil, 9
  seriseq, 23
  seriseq2, 24
  wiepkema, 28
  xcsa, 29
∗Topic **multivariate**
  ca, 5
  compmat, 6
  compseq, 8
  flux, 11
  geneseq, 13
  homoth, 14
  makedico, 15
  makeSeries, 16
  matrcom, 17
  mattrans, 18
  multMatrans, 19
  read.old.seq, 20
  read.series.seq, 21
  rotation, 22
  smartArrow, 25
  symet, 27
∗Topic **package**
  sequence-package, 2

aphelinus, 3, *4*
aphmat, 4
arrows, *27*

ca, *2*, 5, *7*, *9*, *12*, *17*
cmdscale, *7*, *9*
compmat, 6, *8*, *12*
compseq, *2*, *6*, 8, *12*, *14*, *16*, *18*, *21*, *22*
courtil, 9

flux, *2*, *5*, *6*, *9*, 11, *17*, *26–28*

geneseq, 13

hclust, *7*, *9*
homoth, 14

makedico, 15
makeSeries, 16
matrcom, 17, *18*, *28*
mattrans, 18
multMatrans, 19

polygon, *26*, *27*
princomp, *6*

read.old.seq, *16*, 20
read.series.seq, *2*, *12*, *16*, *21*, 21, *24*
rotation, 22

sequence-package, 2
seriseq, 23
seriseq2, 24
smartArrow, *12*, *14*, *15*, *23*, 25
symbols, *26*, *27*
symet, *2*, *5*, *6*, 27

wiepkema, 28

xcsa, 29