# Package ‘tableplot’

February 20, 2015

**Type** Package

**Title** Represents tables as semi-graphic displays

**Description** Description:

**Version** 0.3-5

**Date** 2012-08-20

**Author** Ernest Kwan and Michael Friendly

**Maintainer** Michael Friendly <friendly@yorku.ca>

**Depends** grid

**Suggests** lattice, corrgram, perturb, car, vcd, abind

**License** GPL

**LazyLoad** yes

**LazyData** yes

**Repository** CRAN

**Repository/R-Forge/Project** tableplot

**Repository/R-Forge/Revision** 14

**Date/Publication** 2012-11-03 07:06:50

**NeedsCompilation** no

## R topics documented:

---

`tableplot-package` *Tableplot: Semi-graphic displays of tables and arrays*

---

## Description

A tableplot (Kwan, 2008) is designed as a semi-graphic display in the form of a table with numeric values, but supplemented by symbols with size proportional to cell value(s), and with other visual attributes (shape, color fill, background fill, etc.) that can be used to encode other information essential to direct visual understanding. Three-way arrays, where the last dimension corresponds to levels of a factor for which the first two dimensions are to be compared are handled by superimposing symbols.

The general graphic method was first designed as a visualization method of for exploratory and confirmatory factor analysis results (Kwan, 2008), allowing easy graphic comparison of alternative solutions, rotations, etc. Friendly & Kwan (2009) use this form for a new display of collinearity diagnostics, and Friendly & Kwan (2011) discuss the use of tableplots in a debate on the roles of tables and graphs in statistical presentation.

## Details

| | |
|---|---|
| Package: | tableplot |
| Type: | Package |
| Version: | 0.3-5 |
| Date: | 2012-08-20 |
| License: | GPL |
| LazyLoad: | yes |

The main function is `tableplot`. `tableplot.colldiag` is provided for collinearity diagnostics (Friendly & Kwan, 2009). In future releases, tableplot methods will be provided for other specialized tables.

## Author(s)

Ernest Kwan and Michael Friendly

Maintainer: Michael Friendly <friendly@yorku.ca>

## References

Kwan, E. (2008). Improving Factor Analysis in Psychology: Innovations Based on the Null Hypothesis Significance Testing Controversy. Ph. D. thesis, York University.

Friendly, M. & Kwan, E. (2009). Where's Waldo: Visualizing Collinearity Diagnostics, *The American Statistician*, 63(1), 56-65.

Kwan, E. and Lu, I. R. R. and Friendly, M. (2009). Tableplot: A new tool for assessing precise predictions, *Zeitschrift für Psychologie / Journal of Psychology*, 217, 38-48.

Friendly, M. & Kwan, E. (2011). Comment (Graph people versus table people: Reply to Gelman), *Journal of Computational and Graphical Statistics*, 20 (1), 18-27.

## See Also

`tableplot`, `tableplot.colldiag`

`corrgram` for corrgrams of correlation matrices

`balloonplot` for balloonplots of two-way tables

## Examples

```
# See: demo(psych9)
```

---

`cellgram` *Draw one cell in a tableplot*

---

## Description

Draws a graphic representing one or more values for one cell in a tableplot using shapes whose size is proportional to the cell values.

## Usage

```
cellgram(cell, shape = 0, shape.col = "black", shape.lty = 1, shape.neg = 0, shape.col.neg = "red", sha
```

## Arguments

| | |
|---|---|
| `cell` | Value(s) to be depicted in the table cell |
| `shape` | Shape(s) used to encode the numerical value of `cell`. Any of `0="circle", 1="diamond", 2="square",` |
| `shape.col` | Outline color(s) for the shape(s) |
| `shape.lty` | Outline color(s) for the shape(s) |
| `shape.neg` | Shape(s) used to encode negative values in `cell`. Any of `0="circle", 1="diamond", 2="square", 3='` |
| `shape.col.neg` | |
| `shape.lty.neg` | |
| `cell.fill` | inside color of \|smallest\| shape in a cell |
| `back.fill` | background color of cell |
| `label` | how many cell values will be printed; max is 4 |
| `label.size` | size of cell label(s) |
| `label.col` | color of cell label(s) |
| `ref.lines` | whether to draw ref lines or not |
| `ref.col` | color of reference lines |
| `scale.max` | |
| `shape.lwd` | Width of line used to draw the shape(s) |
| `frame.col` | color of frame around cell |
| `frame.lwd` | line width of frame around cell |

## Value

No value is returned

## Author(s)

Ernest Kwan

## See Also

`tableplot`

---

`compare` *Calculate factor comparison statistics*

---

## Description

Function to facilitate the comparison of two or more factor patterns stored as a 3-dimensional array. If only two patterns, a matrix of differences is calculated. If three of more patterns, a matrix of standard deviations is calculated

## Usage

```
compare(X)
```

## Arguments

| | |
|---|---|
| `X` | A 3 dimensional array, where the last dimension corresponds to different studies or factor solutions. |

## Value

A matrix comparing the factor solutions

## Author(s)

Ernest Kwan

## See Also

`congruence.coef`

## congruence.array — *Congruence coefficients for a 3-way array*

### Description

Calculates congruence coefficients (or some other statistic) for the rows and columns of a three-way array over the last dimension (layers) of the array.

Typically, each layer of the array gives factor or component loadings for a different sample or rotation.

### Usage

```
congruence.array(X, FUN = congruence.coef, stat.name = "phi", round = FALSE, scale = 1, ref = "last")
```

### Arguments

| | |
|---|---|
| `X` | A three-dimensional array |
| `FUN` | A function of two vector arguments, returning a single number |
| `stat.name` | Name for the statistic calculated |
| `round` | If `TRUE`, round the `scaleed` array to the nearest integer. If numeric, the the `scaleed` array is rounded to the given number of decimal places. |
| `scale` | Multiplier for the array returned |
| `ref` | Reference (baseline) level of the third dimension of `X` |

### Value

An array with one more row and column than `X` and the same number of layers, containing the calculated statistic in the last row and column.

### Author(s)

Michael Friendly

### See Also

`congruence.coef`, ~~~

### Examples

```
NEO.sm <- transpose(NEO[1:12,,])
congruence.array(NEO.sm, scale=100, round=TRUE)

congruence.array(NEO.sm, FUN=function(a,b) max(abs(a-b)), stat.name="max.diff", round=2)
```

`Linehan` *Factor patterns for two samples from Linehan et al. (2006)*

### Description

Factor patterns for two samples from Linehan et al. (2006). There are 13 items, 4 factors and two samples, giving a 13 x 4 x 2 array.

### Usage

```
data(Linehan)
```

### Format

The format is:

```
 num [1:13, 1:4, 1:2] 0.864 0.851 0.829 0.803 0.318 -0.218 -0.191 0.146 0.108 -0.154 ...
 - attr(*, "dimnames")=List of 3
  ..$ : NULL
  ..$ : chr [1:4] "SI" "RL" "SC" "L"
  ..$ : NULL
```

### Source

Linehan, M. M. and Comtois, K. A. and Brown, M. Z. and Heard, H. L. and Wagner, A. (2006). Suicide Attempt Self-Injury Interview (SASII): development, reliability, and validity of s scale to assess suicide attempts and intentional self-injury. *Psychological Assessment*, 18(3), 303-312.

### Examples

```
data(Linehan)

# Linehan samples superimposed

tableplot(
values = round(Linehan*100),
assign.sets = matrix(1,13,4),
cell.specs=list(list(0,"grey50",1,1,"red",1,"white","grey90",2,.7,"black",FALSE,"black",99)),
left.space=8, top.space=8 )

# Vectorized arguments
# Use color to distinguish samples

tableplot(
values = round(Linehan*100),
assign.sets = matrix(1,13,4),
cell.specs=list(list(0,c("black","blue"),1,1,c("black","blue"),1,"white","grey80",2,.7,"black",FALSE,"black",9
)
```

```
################################################
# augment the table with congruence coefficients
Linehan.augment <- array(NA, c(14,5,2))
Linehan.augment[1:13,1:4,1:2] <- Linehan

# For rows:
for (i in 1:13){
Linehan.augment[i,5,1] <- congruence.coef(Linehan[i,,1],Linehan[i,,2])
}
# For columns:
for (j in 1:4){
Linehan.augment[14,j,1] <- congruence.coef(Linehan[,j,1],Linehan[,j,2])
}
# For overall:
Linehan.augment[14,5,1] <- congruence.coef(as.vector(Linehan[,,1]),as.vector(Linehan[,,2]))

dimnames(Linehan.augment) <- list(c(1:13,"phi"),c("SI","RL","SC","L","phi"),NULL)

# Difference cell specs for last row & col
M <- matrix(1,14,5)
M[14,] <- 2
M[,5] <- 2

# Linehan superimposed, augmented by congruence coefficients
tableplot(
values = round(Linehan.augment*100),
assign.sets = M,
cell.specs=list(
list(0,"grey50",1,1,"red",1,"white","grey95",2,.75,"black",FALSE,"black",99),
list(0,"grey30",1,1,"red",1,"white","yellow",1,.75,"black",FALSE,"black",100)),
h.parts = c(13,1), v.parts = c(4,1), gap = 1,
left.space=10, top.space=10)
```

---

make.specs *Construct cell specifications for a tableplot*

---

## Description

Construct one or more sets of of cell specifications for a tableplot, using the arguments to `cellgram`, to be used as the `assign.sets` argument in a `tableplot`.

## Usage

```
make.specs(n = NULL, as.data.frame = FALSE, subset, ...)
```

## Arguments

| | |
|---|---|
| n | If specified, determines the number of cell specifications generated. Otherwise, the maximum length of any of the dots argument. |

| | |
|---|---|
| `as.data.frame` | If `TRUE`, returns the result as a data.frame. Mainly for viewing the results in an easier way. |
| `subset` | A list of names of the arguments to `cellgram` to be included in the result. Mainly used for showing simplified examples. |
| `...` | Arguments to `cellgram`, each given as a single element or a vector. Each argument is replicated to the length of the longest one. |

### Details

The function uses `formals(cellgram)[-1]` to retrieve the names of arguments and default values that can be included in `assign.sets`.

### Value

Unless `as.data.frame` is `TRUE`, returns a list of lists of arguments to `cellgram` that can be used as the `assign.sets` argument of `tableplot`.

### Author(s)

Michael Friendly

### See Also

`cellgram`

### Examples

```
# generate 4 sets of cell specifications
specs <- make.specs(
shape=c(0, 0, 0, 2),           #  circles and squares
    cell.fill=c("red","blue","green", "grey40"),
back.fill="white",
scale.max=100
)
```

---

## make.specs0 — *Construct a set of cell specifications for a tableplot*

---

### Description

Construct a set of cell specifications for a tableplot

### Usage

```
make.specs0(n = NULL, as.data.frame = FALSE, shape = 0, shape.col = "black", shape.lty = 1, shape.neg =
cell.fill = "white", back.fill = "white", label = 0, label.size = 0.7, label.col = "black", ref.lines =
```

### Arguments

| | |
|---|---|
| `n` | Number of cell specifications to generate. If not specified, the length of the longest cellgram argument is used. |
| `as.data.frame` | |
| `shape` | |
| `shape.col` | |
| `shape.lty` | |
| `shape.neg` | |
| `shape.col.neg` | |
| `shape.lty.neg` | |
| `cell.fill` | |
| `back.fill` | |
| `label` | |
| `label.size` | |
| `label.col` | |
| `ref.lines` | |
| `ref.col` | |
| `scale.max` | |
| `shape.lwd` | |
| `frame.col` | |
| `frame.lwd` | |

### Details

This function is replaced by `make.specs` and will be removed.

### Author(s)

Michael Friendly

### See Also

`cellgram`

NEO *Factor solutions for the NEO PI-R*

## Description

Factor solutions for the "Big 5" dimensions of personality determined using the Revised NEO Personality Inventory (NEO PI-R; Costa & McCrae, 1992). The five dimensions are measured by 240 items grouped into 30 sub-scales (“facets”), with six facets measuring each of the five dimensions.

`NEO.n` is from the normative sample of Costa & McCrae, 1992. `NEO.s` is from a cross-cultural Shona-speaking sample from Zimbabwe (Piedmont etal., 2002).

## Usage

```
data(NEO.n)
data(NEO.s)
data(NEO)
```

## Format

For `NEO.n`:

The format is:

```
num [1:30, 1:5] 0.81 0.63 0.8 0.73 0.49 0.7 -0.12 -0.18 -0.32 0.04 ...
- attr(*, "dimnames")=List of 2
  ..$ : chr [1:30] "N1" "N2" "N3" "N4" ...
  ..$ : chr [1:5] "N" "E" "O" "A" ...
```

For `NEO.s`:

The format is:

```
num [1:30, 1:5] 0.66 0.53 0.6 0.58 0.58 0.57 -0.24 -0.14 -0.51 -0.15 ...
- attr(*, "dimnames")=List of 2
  ..$ : chr [1:30] "N1" "N2" "N3" "N4" ...
  ..$ : chr [1:5] "N" "E" "O" "A" ...
```

The `NEO` data is the three-way array combining `NEO.n` and `NEO.s`: The format is:

```
num [1:30, 1:5, 1:2] 0.81 0.63 0.8 0.73 0.49 0.7 -0.12 -0.18 -0.32 0.04 ...
- attr(*, "dimnames")=List of 3
  ..$ : chr [1:30] "N1" "N2" "N3" "N4" ...
  ..$ : chr [1:5] "N" "E" "O" "A" ...
  ..$ : chr [1:2] "Normative" "Shona"
```

## Source

Costa Jr, P. T. & McCrae, R. R. (1992). Normal personality assessment in clinical practice: The NEO Personality Inventory *Psychological Assessment*, 4, 5-13.

Piedmont, R. L. and Bain, E. and McCrae, R.R. and Costa Jr, P. T.(2002). "The applicability of the five-factor model in a sub-Saharan culture: The NEO PI-R in Shona", In R. R. McCrae and J. Allik (ed.) *The Five-Factor Model of Personality Across Cultures*, New York: Kluwer Academic/Plenum, 155-173.

## References

Kwan, E. and Lu, I. R. R. and Friendly, M. (2009). Tableplot: A new tool for assessing precise predictions *Zeitschrift für Psychologie / Journal of Psychology*, 217, 38-48.

## Examples

```
data(NEO.n); data(NEO.s)

# Examples from Kwan et al., 2009

# Plot of Normative patter, first 12 facets:
tableplot(
values = round( 100 * t(NEO.n[1:12,])),
label.size = 1.5,
cell.specs=list(
list(0,"blue",1,1,"red",1,"white","grey90",1,1.5,"grey50",FALSE,"grey40",100)),
v.parts = c(6,6),
gap = 3,
left.space=15,
top.space=15,
assign.sets = matrix(1,5,12))


facnames <- c("N","E","O","A","C")
itmnames <- as.vector(t(outer(facnames, 1:6, paste, sep="")))

# Put the patterns together:
  neopir <- array(NA, c(6,31,2))
  neopir[1:5,1:30,1] <- t(NEO.n) # Normative
  neopir[1:5,1:30,2] <- t(NEO.s) # Shona

# Calculate congruence coefficients for variables:
  for (j in 1:30){
neopir[6,j,] <- round(congruence.coef(neopir[1:5,j,1],neopir[1:5,j,2]),2) }

# Calculate congruence coefficients for factors:
  for (i in 1:5){
neopir[i,31,] <- round(congruence.coef(neopir[i,1:30,1],neopir[i,1:30,2]),2) }

# Plug in the total congruence coefficient:
  neopir[6,31,] <- 0.89
```

```
# Get rid of decimals:
  neopir <- round(neopir * 100)

 dimnames(neopir) <- list( c(facnames, "phi"), c(itmnames, "phi"), c("Normative", "Shona"))

# Plot of Normative and Shona, superimposed and augmented:
B       <- matrix(1,6,31)
B[6,]   <- 2
B[,31]  <- 2

tableplot(
values = neopir,
label.size = 0.8,
cell.specs=list(
list(0,"blue",1,1,"red",1,"white","grey95",2,0.6,"grey50",FALSE,"grey40",100),
list(0,"blue",1,1,"red",1,"yellow","grey60",1,0.6,"grey10",FALSE,"grey40",100)),
v.parts = c(6,6,6,6,6,1),
h.parts = c(5,1),
gap = 1,
left.space=8,
assign.sets = B)
```

---

tableplot *Tableplot*

---

## Description

A tableplot (Kwan, 2008) is designed as a semi-graphic display in the form of a table with numeric values, but supplemented by symbols with size proportional to cell value(s), and with other visual attributes (shape, color fill, background fill, etc.) that can be used to encode other information essential to direct visual understanding. Three-way arrays, where the last dimension corresponds to levels of a factor for which the first two dimensions are to be compared are handled by superimposing symbols.

## Usage

```
tableplot(values, ...)

## Default S3 method:
tableplot(values, assign.sets, cell.specs,
v.parts = 0, h.parts = 0, gap = 2, text.m = 0, empty.text.size = 0.8, empty.text.col = "grey30",
title = NULL, table.label = TRUE, label.size = 0.8, side.rot = 0, left.space = 10, top.space = 10+10*(!
```

## Arguments

| | |
|---|---|
| `values` | A matrix or 3-dimensional array of values to be displayed in a tableplot |
| `...` | Arguments passed down to `tableplot.default` |

| | |
|---|---|
| `assign.sets` | Matrix of specification assignments, of the same size as the first two dimensions of `values`. Entries refer to the sub-lists of `cell.specs`. Defaults to `matrix(1, dim(values)[1], dim(values)[2]))`. |
| `cell.specs` | List of lists; each list is one specification for the arguments to `cellgram`. `cell.specs[[k]]` is used for all table cells where `assign.sets[i,j] = k`. See `make.specs` for a simple way to construct the `cell.specs` argument. |
| `v.parts` | An integer vector giving the number of columns in two or more partititions of the table. If provided, sum must equal number of columns. |
| `h.parts` | An integer vector giving the number of rows in two or more partititions of the table. If provided, sum must equal number of columns. |
| `gap` | Width of the gap in each partition, if partitions are requested by `v.parts` and/or `h.parts` |
| `text.m` | Matrix of text for insertion into text-only, empty cell(s) |
| `empty.text.size` | Text size for text-only cells |
| `empty.text.col` | Text color for text-only cells |
| `title` | Main title |
| `table.label` | Logical value: whether to print row/column labels. |
| `label.size` | Character size for labels |
| `side.rot` | Degree of rotation (positive for counter-clockwise) |
| `left.space` | Space between left of tableplot and left edge of drawing region, in mm. |
| `top.space` | Space between top of tableplot and top edge of drawing region, in mm. |

## Value

None. Used for its side effect

## Author(s)

Ernest Kwan

## References

Kwan, E. (2008). Improving Factor Analysis in Psychology: Innovations Based on the Null Hypothesis Significance Testing Controversy. Ph. D. thesis, York University.

Kwan, E. and Lu, I. R. R. and Friendly, M. (2009). Tableplot: A new tool for assessing precise predictions *Zeitschrift für Psychologie / Journal of Psychology*, 217, 38-48.

## See Also

`cellgram`, `make.specs`

## Examples

```
# Factor pattern matrix from Nisenbaum etal. (2004)
Nisenbaum <- matrix(c(
 93,  14, -14,  -3,
 87,  17, -13, -12,
 39, -15,  15,  29,
 25,  10,  21,  41,
 36, -10,   8,  35,
  5,  79,  -1,   0,
 10,  72, -11,  16,
  8,  80,  12,  -7,
 23,  47,   6,   8,
-28,  27,   5,  78,
  6,  33, -22,  37,
-10,  21, -10,  68,
-19,  -1,  93,  11,
-11,  -8,  86,   3,
  2,  -1,  53,   3,
 10,  27,  51, -11,
 21,  28,  50, -12,
 18,  10,  11,  23,
  0,  25,  15,  20,
 15, -24,   8,  54,
-13,  31,  14,  20), 21, 4, byrow=TRUE)
colnames(Nisenbaum) = paste("F", 1:4, sep="")

tableplot(
values = Nisenbaum,
cell.specs = list(list(0,"grey50",1,0,"red",1,"white","grey80",1,0.7,"black",FALSE,"black",93)),
assign.sets = matrix(1,21,4)
)
```

---

`tableplot.colldiag` *Tableplot for Collinearity Diagnostics*

---

## Description

Produces a tableplot of collinearity diagnostics for a linear regression model (Friendly & Kwan, 2009), showing condition indices and variance proportions for the quantitative predictors. The goal is to highlight the variables involved in one or more nearly collinear relations among the predictors.

The default scheme is to show the column of condition indices at the left, using color to indicate danger (red), warning (yellow) and OK (green) with colors designed to reproduce as ordered in B/W. The variance proportions for the predictors are shown in a block at the right, using white, pink, red for small, medium and large values.

## Usage

```
## S3 method for class 'colldiag'
tableplot(values, cell.specs,
prop.col = c("white", "pink", "red"),
cond.col = c("#A8F48D", "#DDAB3E", "red"),
cond.max = 100,
prop.breaks = c(0, 20, 50, 100),
cond.breaks = c(0, 5, 10, 1000),
show.rows = nvar:1, ...)
```

## Arguments

| | |
|---|---|
| `values` | A `colldiag` object, such as calculated by `colldiag` |
| `cell.specs` | Specifications for `cellgram` arguments, used only to override those calculated internally from the following arguments. |
| `prop.col` | A vector of colors used to display the values of the variance proportions. |
| `cond.col` | A vector of colors used to display the values of the condition indices |
| `cond.max` | Maximum value for a condition index displayed. |
| `prop.breaks` | Breaks for the variance proportions. |
| `cond.breaks` | Breaks for the condition indices |
| `show.rows` | Vector of indices of the rows of the `colldiag` object to be displayed in the tableplot. By default, all rows are shown, in reverse order, with the highest condition indices at the top. |
| `...` | Other arguments to pass down to `tableplot.default` |

## Details

The values of variance proportions are multiplied by 100 and rounded.

## Value

None. Used for its side-effect.

## Author(s)

Michael Friendly

## References

Friendly, M. & Kwan, E. (2009). Where's Waldo: Visualizing Collinearity Diagnostics *The American Statistician*, 63(1), 56-65.

## See Also

`colldiag` for calculation of collinearity diagnostics

## Examples

```
# Baseball data example, from Friendly & Kwan (2009)
if (require(vcd) && require(perturb)) {
# model, with transformed variables
Baseball$logsal <- log(Baseball$sal87)
Baseball$years7 <- pmin(Baseball$years,7)

base.mod <- lm(logsal ~ years+atbat+hits+homeruns+runs+rbi+walks, data=Baseball)
if (require(car)) {
# examine variance inflation factors
vif(base.mod)
}
# corresponds to SAS: / collinoint option
cd <- colldiag(base.mod, add.intercept=FALSE, center=TRUE)
# simplified display
print(cd, fuzz=.3)

tableplot(cd)
}
```

---

`utility` *Utility functions for tableplots*

---

## Description

Utility functions for producing tableplots

## Usage

```
congruence.coef(a, b)
cg.cf(a, b)
identity.coef(a, b)
id.cf(a, b)
transpose(x)
```

## Arguments

| | |
|---|---|
| a | A vector of factor or component loadings |
| b | A vector of factor or component loadings |
| x | A matrix or array of 3 dimensions. |

## Details

Congruence coefficients (Burt, 1948; Wrigley & Newhaus, 1955) are used to assess the similarity of two rows or columns in a factor pattern....

`transpose` transposes an array of 2 or 3 dimensions, where, in the 3D case, transposition is carried out only on the first two dimensions. Useful to transpose the data for a tableplot.

## Author(s)

Ernest Kwan

## References

Burt, C. L. (1948). The factorial study of temperamental traits. *British Journal of Psychology*, 48, 378-399.

Wrigley, C. C. and Newhaus, J. O. (1955). The matching of two sets of factors. *The American Psychologist*, 10, 418-419.

# Index