User manual for the psych package Version 1.0.63

William Revelle
Department of Psychology
Northwestern University

January 24, 2009

Contents

List of Figures

psych *A package for personality, psychometric, and psychological research*

Description

Overview of the psych package.

The psych package has been developed at Northwestern University to include functions most useful for personality and psychological research. Some of the functions (e.g., `read.clipboard`, `describe`, `pairs.panels`, `error.bars`) are useful for basic data entry and descriptive analyses.

Psychometric applications include routines for Very Simple Structure (`VSS`), Minimum Average Partial correlation (`MAP`), Item Cluster Analysis (`ICLUST`) and principal axes factor analysis (`factor.pa`), as well as functions to do Schmid Leiman transformations (`schmid`) to transform a hierarchical factor structure into a bifactor solution, and to calculate reliability coefficients alpha (`score.items`, `score.multiple.choice`), beta (`ICLUST`) and McDonald's omega (`omega` and `omega.graph`). Guttman's six estimates of internal consistency reliability (`guttman`) and the six measures of Intraclass correlation coefficients (`ICC`) discussed by Shrout and Fleiss are also available.

The `score.items`, and `score.multiple.choice` functions may be used to form single or multiple scales from sets of dichotomous, multilevel, or multiple choice items by specifying scoring keys.

Additional functions make for more convenient descriptions of item characteristics. Functions under development include 1 and 2 parameter Item Response measures.

A number of procedures have been developed as part of the Synthetic Aperture Personality Assessment (SAPA) project. These routines facilitate forming and analyzing composite scales equivalent to using the raw data but doing so by adding within and between cluster/scale item correlations. These functions include extracting clusters from factor loading matrices (`factor2cluster`), synthetically forming clusters from correlation matrices (`cluster.cor`), and finding multiple ((`mat.regress`) and partial ((`partial.r`) correlations from correlation matrices.

Functions to generate simulated data with particular structures include `sim.circ` (for circumplex structures), `sim.item` (for general structures) and `sim.congeneric` (for a specific demonstration of congeneric measurement). The functions `sim.congeneric` and `sim.hierarchical` can be used to create data sets with particular structural properties. A more general form for all of these is `sim.structural` for generating general structural models.

Functions to apply various standard statistical tests include `p.rep` and its variants for testing the probability of replication, `r.con` for the confidence intervals of a correlation, `r.test` to test single, paired, or sets of correlations.

In order to study diurnal or circadian variations in mood, it is helpful to use circular statistics. Functions to find the circular mean (`circadian.mean`), circular (phasic) correlations (`circadian.cor`) and the correlation between linear variables and circular variables (`circadian.linear.cor`) supplement a function to find the best fitting phase angle (`cosinor`) for measures taken with a fixed period (e.g., 24 hours).

The most recent development version of the package is always available for download as a *source* file from the repository at `http://personality-project.org/r/src/contrib/`.

Details

The psych package was originally a combination of multiple source files maintained at the `http://personality-project.org/r` repository: "useful.r", VSS.r., ICLUST.r, omega.r, etc."useful.r" is a set of routines for easy data entry (`read.clipboard`), simple descriptive statistics (`describe`), and splom plots combined with correlations (`pairs.panels`, adapted from the help files of pairs). It is now a single package.

The `VSS` routines allow for testing the number of factors (`VSS`), showing plots (`VSS.plot`) of goodness of fit, and basic routines for estimating the number of factors/components to extract by using the `MAP`'s procedure, the examining the scree plot (`VSS.scree`) or comparing with the scree of an equivalent matrix of random numbers (`VSS.parallel`).

In addition, there are routines for hierarchical factor analysis using Schmid Leiman tranformations (`omega`, `omega.graph`) as well as Item Cluster analysis (`ICLUST`, `ICLUST.graph`).

The more important functions in the package are for the analysis of multivariate data, with an emphasis upon those functions useful in scale construction of item composites.

When given a set of items from a personality inventory, one goal is to combine these into higher level item composites. This leads to several questions:

1) What are the basic properties of the data? `describe` reports basic summary statistics (mean, sd, median, mad, range, minimum, maximum, skew, kurtosis, standard error) for vectors, columns of matrices, or data.frames. `describe.by` provides descriptive statistics, organized by one or more grouping variables. `pairs.panels` shows scatter plot matrices (SPLOMs) as well as histograms and the Pearson correlation for scales or items. `error.bars` will plot variable means with associated confidence intervals. `error.bars` will plot confidence intervals for both the x and y coordinates.

2) What is the most appropriate number of item composites to form? After finding either standard Pearson correlations, or finding tetrachoric or polychoric correlations using a wrapper (`poly.mat`) for John Fox's hetcor function, the dimensionality of the correlation matrix may be examined. The number of factors/components problem is a standard question of factor analysis, cluster analysis, or principal components analysis. Unfortunately, there is no agreed upon answer. The Very Simple Structure (`VSS`) set of procedures has been proposed as on answer to the question of the optimal number of factors. Other procedures (`VSS.scree`, `VSS.parallel`, `fa.parallel`, and `MAP`) also address this question.

3) What are the best composites to form? Although this may be answered using principal components (`principal)` or principal axis factor analysis (`factor.pa)` and to show the results graphically (`fa.graph)`, it is sometimes more useful to address this question using cluster analytic techniques. (Some would argue that better yet is to use maximum likelihood factor analysis using `factanal` from the stats package.) Previous versions of `ICLUST` (e.g., Revelle, 1979) have been shown to be particularly successful at doing this. Graphical output from `ICLUST.graph` uses the Graphviz dot language and allows one to write files suitable for Graphviz.

Graphical organizations of cluster and factor analysis output can be done using `cluster.plot` which plots items by cluster/factor loadings and assigns items to that dimension with the highest loading.

4) How well does a particular item composite reflect a single construct? This is a question of reliability and general factor saturation. Multiple solutions for this problem result in (Cronbach's) alpha (`score.items`), (Revelle's) Beta (`ICLUST`), and (McDonald's) `omega` (both omega hierarchical and omega total). Additional reliability estimates may be found in the `guttman` function.

5) For some applications, data matrices are synthetically combined from sampling different items for different people. So called Synthetic Aperture Personality Assessement (SAPA) techniques allow the formation of large correlation or covariance matrices even though no one person has taken all of the items. To analyze such data sets, it is easy to form item composites based upon the covariance matrix of the items, rather than original data set. These matrices may then be analyzed using a number of functions (e.g., `cluster.cor`, `factor.pa`, `ICLUST`, `principal` , `mat.regress`, and `factor2cluster`.

6) More typically, one has a raw data set to analyze. `score.items` will score data sets on multiple scales, reporting the scale scores, item-scale and scale-scale correlations, as well as coefficient alpha and alpha-1. Using a 'keys' matrix (created by `make.keys` or by hand), scales can have overlapping or independent items. `score.multiple.choice` scores multiple choice items or converts multiple choice items to dichtomous (0/1) format for other functions.

An additional set of functions generate simulated data to meet certain structural properties. `item.sim` creates simple structure data, `sim.circ` will produce circumplex structured data, `item.dichot` produces circumplex or simple structured data for dichotomous items. These item structures are useful for understanding the effects of skew, differential item endorsement on factor and cluster analytic soutions.

When examining personality items, some people like to discuss them as representing items in a two dimensional space with a circumplex structure. Tests of circumplex fit `circ.tests` have been developed. When representing items in a circumplex, it is convenient to view them in `polar` coordinates.

Additional functions for testing the difference between two independent or dependent correlation `r.test`, to find the `phi` or `Yule` coefficients from a two by table, or to find the confidence interval of a correlation coefficient.

Ten data sets are included: `bfi` represents 25 personality items thought to represent five factors of personality, `iqitems` has 14 multiple choice iq items. `sat.act` has data on self reported test scores by age and gender. `galton` Galton's data set of the heights of parents and their children. `peas` recreates the original Galton data set of the genetics of sweet peas. `heights` and `cubits` provide even more Galton data, `vegetables` provides the Guilford preference matrix of vegetables. `cities` provides airline miles between 11 US cities (demo data for multidimensional scaling).

Package:	psych
Type:	Package
Version:	1.0-63
Date:	2009-1-15
License:	GPL version 2 or newer

Index:

psych A package for personality, psychometric, and psychological research.

Useful data entry and descriptive statistics

read.clipboard	shortcut for reading from the clipboard
read.clipboard.csv	shortcut for reading comma delimited files from clipboard
read.clipboard.lower	shortcut for reading lower triangular matrices from the clipboard
read.clipboard.upper	shortcut for reading upper triangular matrices from the clipboard
describe	Basic descriptive statistics useful for psychometrics
describe.by	Find summary statistics by groups
headtail	combines the head and tail functions for showing data sets
pairs.panels	SPLOM and correlations for a data matrix
multi.hist	Histograms and densities of multiple variables arranged in matrix form
skew	Calculate skew for a vector, each column of a matrix, or data.frame
kurtosi	Calculate kurtosis for a vector, each column of a matrix or dataframe
geometric.mean	Find the geometric mean of a vector or columns of a data.frame
harmonic.mean	Find the harmonic mean of a vector or columns of a data.frame
error.bars	Plot means and error bars
error.bars.by	Plot means and error bars for separate groups
error.crosses	Two way error bars
interp.median	Find the interpolated median, quartiles, or general quantiles.
rescale	Rescale data to specified mean and standard deviation
table2df	Convert a two dimensional table of counts to a matrix or data frame

Data reduction through cluster and factor analysis

factor.pa	Do a principal Axis factor analysis
fa.graph	Show the results of a factor analysis or principal components analysis graphically
principal	Do an eigen value decomposition to find the principal components of a matrix
fa.parallel	Scree test and Parallel analysis
guttman	8 different measures of reliability (6 from Guttman (1945)
`ICLUST`	Apply the ICLUST algorithm
ICLUST.graph	Graph the output from ICLUST using the dot language
ICLUST.rgraph	Graph the output from ICLUST using rgraphviz
poly.mat	Find the polychoric correlations for items (uses J. Fox's hetcor
omega	Calculate the omega estimate of factor saturation (requires the GPArotation package)
omega.graph	Draw a hierarchical or Schmid Leiman orthogonalized solution
schmid	Apply the Schmid Leiman transformation to a correlation matrix
score.items	Combine items into multiple scales and find alpha
score.multiple.choice	Combine items into multiple scales and find alpha and basic scale statistics
smc	Find the Squared Multiple Correlation (used for initial communality estimates)
VSS	Apply the Very Simple Structure criterion to determine the appropriate number of factors.
VSS.parallel	Do a parallel analysis to determine the number of factors for a random matrix
VSS.plot	Plot VSS output
VSS.scree	Show the scree plot of the factor/principal components
MAP	Apply the Velicer Minimum Absolute Partial criterion for number of factors

Procedures particularly useful for Synthetic Aperture Personality Assessment

alpha.scale	Find coefficient alpha for a scale (see also score.items)
make.keys	Create the keys file for score.items or cluster.cor
correct.cor	Correct a correlation matrix for unreliability
count.pairwise	Count the number of complete cases when doing pair wise correlations
cluster.cor	find correlations of composite variables from larger matrix
cluster.loadings	find correlations of items with composite variables from a larger matrix
eigen.loadings	Find the loadings when doing an eigen value decomposition
factor.pa	Do a Principal Axis factor analysis and estimate factor scores
factor2cluster	extract cluster definitions from factor loadings
factor.congruence	Factor congruence coefficient
factor.fit	How well does a factor model fit a correlation matrix
factor.model	Reproduce a correlation matrix based upon the factor model
factor.residuals	Fit = data - model
factor.rotate	"hand rotate" factors
guttman	8 different measures of reliability
mat.regress	standardized multiple regression from raw or correlation matrix input
principal	Do an eigen value decomposition to find the principal components of a matrix

Functions for generating simulated data sets

sim.circ	Generate a two dimensional circumplex item structure
sim.item	Generate a two dimensional simple structrue with particular item characteristics
sim.congeneric	Generate a one factor congeneric reliability structure
sim.structural	Generate a multifactorial structural model
sim.VSS	Generate simulated data for the factor model
phi.demo	Create artificial data matrices for teaching purposes
sim.hierarchical	Generate simulated correlation matrices with hierarchical or any structure

Graphical functions (require Rgraphviz)

structure.graph	Draw a sem or regression graph
fa.graph	Draw the factor structure from a factor or principal components analysis
omega.graph	Draw the factor structure from an omega analysis (either with or without the Schmid Leiman tra
ICLUST.graph	Draw the tree diagram from ICLUST

Circular statistics (for circadian data analysis)

circadian.cor
circadian.linear.cor
circadian.mean
cosinor

Miscellaneous functions

comorbidity	Convert base rate and comorbity to phi, Yule and tetrachoric
fisherz	Apply the Fisher r to z transform
fisherz2r	Apply the Fisher z to r transform
ICC	Intraclass correlation coefficients
cortest.mat	Test for equality of two matrices (see also cortest.normal, cortest.jennrich)
cortest.bartlett	Test whether a matrix is an identity matrix
paired.r	Test for the difference of two paired or two independent correlations
r.con	Confidence intervals for correlation coefficients
r.test	Test of significance of r, differences between rs.
p.rep	The probability of replication given a p, r, t, or F
phi	Find the phi coefficient of correlation from a 2 x 2 table
phi.demo	Demonstrate the problem of phi coefficients with varying cut points
phi2poly	Given a phi coefficient, what is the polychoric correlation
phi2poly.matrix	Given a phi coefficient, what is the polychoric correlation (works on matrices)
polar	Convert 2 dimensional factor loadings to polar coordinates.
poly.mat	Use John Fox's hetcor to create a matrix of correlations from a data.frame or matrix of integer va
polychor.matrix	Use John Fox's polycor to create a matrix of polychoric correlations from a matrix of Yule correla
scaling.fits	Compares alternative scaling solutions and gives goodness of fits
thurstone	Thurstone Case V scaling
tr	Find the trace of a square matrix
wkappa	weighted and unweighted versions of Cohen's kappa
Yule	Find the Yule Q coefficient of correlation
Yule.inv	What is the two by two table that produces a Yule Q with set marginals?
Yule2phi	What is the phi coefficient corresponding to a Yule Q with set marginals?
Yule2phi.matrix	Convert a matrix of Yule coefficients to a matrix of phi coefficients.
Yule2phi.matrix	Convert a matrix of Yule coefficients to a matrix of polychoric coefficients.

Functions that are under development and not recommended for casual use

irt.item.diff.rasch	IRT estimate of item difficulty with assumption that theta = 0
irt.person.rasch	Item Response Theory estimates of theta (ability) using a Rasch like model

Data sets included in the psych package

bfi	represents 25 personality items thought to represent five factors of personality
cities	The airline distances between 11 cities (used to demonstrate MDS)
epi.bfi	13 personality scales
iqitems	14 multiple choice iq items
sat.act	Self reported ACT and SAT Verbal and Quantitative scores by age and gender
galton	Galton's data set of the heights of parents and their children
heights	Galton's data set of the relationship between height and forearm (cubit) length

cubits	Galton's data table of height and forearm length
peas	Galton's data set of the diameters of 700 parent and offspring sweet peas
vegetables	Guilford's preference matrix of vegetables (used for thurstone)

A debugging function that may also be used as a demonstration of psych.

test.psych Run a test of the major functions on 5 different data sets. Primarily for development purposes. Althou

Note

Development versions (source code) of this package are maintained at the repository `http://personality-project.org/r` along with further documentation. Specify that you are downloading a source package.
Some functions require other packages. Specifically, omega and schmid require the GPArotation package, and poly.mat, phi2poly and polychor.matrix requires John Fox's polychor package. ICLUST.rgraph and fa.graph require Rgraphviz. i.e.:

function	requires
omega	GPArotation
schmid	GPArotation
poly.mat	polychor
phi2poly	polychor
polychor.matrix	polychor
ICLUST.rgraph	Rgraphviz
fa.graph	Rgraphviz
structure.graph	Rgraphviz

Author(s)

William Revelle
Department of Psychology
Northwestern University
Evanston, Illiniois
`http://personality-project.org/revelle.html`

Maintainer: William Revelle <revelle@northwestern.edu>

References

A general guide to personality theory and research may be found at the personality-project `http://personality-project.org`. See also the short guide to R at `http://personality-project.org/r`. In addition, see

Revelle, W. (in preparation) An Introduction to Psychometric Theory with applications in R. Springer. at `http://personality-project.org/r/book/`

Examples

```
#See the separate man pages
```

```
test.psych()
```

`alpha.scale` *Cronbach alpha for a scale*

Description

Find Cronbach's coefficient alpha given a scale and a data.frame of the items in the scale. For X, a total score composed of items in the data.frame Y, find Cronbach's alpha.

Usage

```
alpha.scale(x, y)
```

Arguments

`x`	A scale made by summing items
`y`	A data frame of items

Details

Alpha is one of several estimates of the internal consistency reliability of a test. Perhaps because it is so easy to calculate, it is without doubt the most frequently reported measure of internal consistency reliability. Alpha is the mean of all possible spit half reliabilities (corrected for test length). For a unifactorial test, it is a reasonable estimate of the first factor saturation, although if the test has any microstructure (i.e., if it is "lumpy") coefficients beta (see `ICLUST` and `link{omega}` are more appropriate estimates of the general factor saturation.

Alpha is a positive function of the number of items in a test as well as the average intercorrelation of the items in the test. When calculated from the item variances and total test variance, as is done here, alpha is sensitive to differences in the item variances. Alternative functions `score.items` and `cluster.cor` will also score multiple scales and report more useful statistics. "Standardized" alpha is calculated from the inter-item correlations and will differ from raw alpha. Standardized alpha can be found by using `cluster.cor`.

Value

Coefficient alpha

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu⟩

References

`http://personality-project.org/revelle/syllabi/405.syllabus.html`

See Also

score.items, cluster.cor, ICLUST, omega, describe,pairs.panels, alpha in psychometrics

Examples

```
y <- attitude     #from the datasets package
x <- rowSums(y)   #find the sum of the seven attitudes
alpha.scale(x,y)
#[1] 0.8431428
#compare with standardized alpha:
st.alpha <- cluster.cor(rep(1,7),cor(attitude),digits=4)
st.alpha
#compare with score.items
si <- score.items(rep(1,7), attitude,digits=3)
si$alpha
```

---

bfi *25 Personality items representing 5 factors*

Description

25 personality self report items taken from the International Personality Item Pool (ipip.ori.org) were included as part of the Synthetic Aperture Personality Assessment (SAPA) web based personality assessment project. The data from 1000 subjects are included here as a demonstration set for scale construction and factor analysis.

Usage

```
data(bfi)
```

Format

A data frame with 1000 observations on the following 25 variables.

A1 Am indifferent to the feelings of others.

A2 Inquire about others' well-being.

A3 Know how to comfort others.

A4 Love children.

A5 Make people feel at ease.

C1 Am exacting in my work.

C2 Continue until everything is perfect.

C3 Do things according to a plan.

C4 Do things in a half-way manner.

C5 Waste my time.

E1 Don't talk a lot.

E2 Find it difficult to approach others.

E3 Know how to captivate people.

E4 Make friends easily.

E5 Take charge.

N1 Get angry easily.

N2 Get irritated easily.

N3 Have frequent mood swings.

N4 Often feel blue.

N5 Panic easily.

O1 Am full of ideas.

O2 Avoid imposing my will on others.

O3 Carry the conversation to a higher level.

O4 Spend time reflecting on things.

O5 Will not probe deeply into a subject.

Details

The 25 items are organized by five putative factors: Agreeableness, Conscientiousness, Extraversion, Neuroticism, and Openness. The scoring key is created using `make.keys`, the scores are found using `score.items`

Source

The items are from the ipip (Goldberg, 1999). The data are from the SAPA project (Revelle, Wilt and Rosenthal, 2009) , collected Fall, 2006.

References

Goldberg, L.R. (1999) A broad-bandwidth, public domain, personality inventory measuring the lower-level facets of several five-factor models. In Mervielde, I. and Deary, I. and De Fruyt, F. and Ostendorf, F. (eds) Personality psychology in Europe. 7. Tilburg University Press. Tilburg, The Netherlands.

Revelle, W., Wilt, J., and Rosenthal, A. (2009) Personality and Cognition: The Personality-Cognition Link. In Gruszka, A. and Matthews, G. and Szymura, B. (Eds.) Handbook of Individual Differences in Cognition: Attention, Memory and Executive Control, Springer.

Examples

```
data(bfi)
describe(bfi)
 data(bfi)
 keys.list <- list(Agree=c(-1,2:5),Conscientious=c(6:8,-9,-10),Extraversion=c(-11,-12,13:15),Neuroticism=
 keys <- make.keys(25,keys.list,item.labels=colnames(bfi))
```

```
scores <- score.items(keys,bfi,short=TRUE)
scores
```

`bifactor` *Two data sets showing a bifactor solution.*

Description

Holzinger-Swineford (1937) introduced the bifactor model of a general factor and uncorrelated group factors. The Holzinger correlation matrix is a 14 * 14 matrix from their paper. The Reise data set is 16 * 16 correlation matrix of mental health items.

Usage

```
data(bifactor)
```

Details

Holzinger and Swineford introduced the bifactor model (one general factor and several group factors) for mental abilities. This is a nice demonstration data set of a hierarchical factor structure that can be analyzed using the `omega` function or using sem. The bifactor model is typically used in measures of cognitive ability.

The 14 variables are ordered to reflect 3 spatial tests, 3 mental speed tests, 4 motor speed tests, and 4 verbal tests.

More recent applications are to the measurement of psychological status. The Reise data set is a correlation matrix based upon >35,000 observations to the Consumer Assessment of Health Care Provideers and Systems survey instrument. Reise, Morizot, and Hays (2007) describe a bifactor solution based upon 1,000 cases.

The five factors from Reise et al. reflect Getting care quickly (1-3), Doctor communicates well (4-7), Courteous and helpful staff (8,9), Getting needed care (10-13), and and Health plan customer service (14-16).

Source

Holzinger: Holzinger and Swineford (1937)
Reise: Steve Reise (personal communication)

References

Holzinger, Karl and Swineford, Frances (1937) The Bi-factor method. Psychometrika, 2, 41-54

Reise, Steven and Morizot, Julien and Hays, Ron (2007) The role of the bifactor model in resolving dimensionality issues in health outcomes measures. Quality of Life Research. 16, 19-31.

Examples

```
data(bifactor)
holz <- omega(Holzinger,4, title = "14 ability tests from Holzinger-Swineford")
bf <- omega(Reise,5,title="16 health items from Reise")
omega(Reise,5,labels=colnames(Reise),title="16 health items from Reise")
```

---

circ.tests *Apply four tests of circumplex versus simple structure*

Description

Rotations of factor analysis and principal components analysis solutions typically try to represent correlation matrices as simple structured. An alternative structure, appealing to some, is a circumplex structure where the variables are uniformly spaced on the perimeter of a circle in a two dimensional space. Generating these data is straightforward, and is useful for exploring alternative solutions to affect and personality structure.

Usage

```
circ.tests(loads, loading = TRUE, sorting = TRUE)
```

Arguments

`loads`	A matrix of loadings `loads` here
`loading`	Are these loadings or a correlation matrix `loading`
`sorting`	Should the variables be sorted `sorting`

Details

"A common model for representing psychological data is simple structure (Thurstone, 1947). According to one common interpretation, data are simple structured when items or scales have non-zero factor loadings on one and only one factor (Revelle & Rocklin, 1979). Despite the commonplace application of simple structure, some psychological models are defined by a lack of simple structure. Circumplexes (Guttman, 1954) are one kind of model in which simple structure is lacking.

"A number of elementary requirements can be teased out of the idea of circumplex structure. First, circumplex structure implies minimally that variables are interrelated; random noise does not a circumplex make. Second, circumplex structure implies that the domain in question is optimally represented by two and only two dimensions. Third, circumplex structure implies that variables do not group or clump along the two axes, as in simple structure, but rather that there are always interstitial variables between any orthogonal pair of axes (Saucier, 1992). In the ideal case, this quality will be reflected in equal spacing of variables along the circumference of the circle (Gurtman, 1994; Wiggins, Steiger, & Gaelick, 1981). Fourth, circumplex structure implies that variables have a constant radius from the center of the circle, which implies that all variables have equal communality on the two circumplex dimensions (Fisher, 1997; Gurtman, 1994). Fifth, circumplex structure

implies that all rotations are equally good representations of the domain (Conte & Plutchik, 1981; Larsen & Diener, 1992). (Acton and Revelle, 2004)

Acton and Revelle reviewed the effectiveness of 10 tests of circumplex structure and found that four did a particularly good job of discriminating circumplex structure from simple structure, or circumplexes from ellipsoidal structures. Unfortunately, their work was done in Pascal and is not easily available. Here we release R code to do the four most useful tests:

1 The Gap test of equal spacing

2 Fisher's test of equality of axes

3 A test of indifference to Rotation

4 A test of equal Variance of squared factor loadings across arbitrary rotations.

To interpret the values of these various tests, it is useful to compare the particular solution to simulated solutions representing pure cases of circumplex and simple structure. See the example output from `circ.simulation` and compare these plots with the results of the circ.test.

Value

A list of four items is returned. These are the gap, fisher, rotation and variance test results.

`gaps`	gap.test
`fisher`	fisher.test
`RT`	rotation.test
`VT`	variance.test

Note

Of the 10 criterion discussed in Acton and Revelle (2004), these tests operationalize the four most useful.

Author(s)

William Revelle

References

Acton, G. S. and Revelle, W. (2004) Evaluation of Ten Psychometric Criteria for Circumplex Structure. Methods of Psychological Research Online, Vol. 9, No. 1 `http://personality-project.org/revelle/publications/acton.revelle.mpr110_10.pdf`

See Also

`circ.simulation`, `sim.circ`

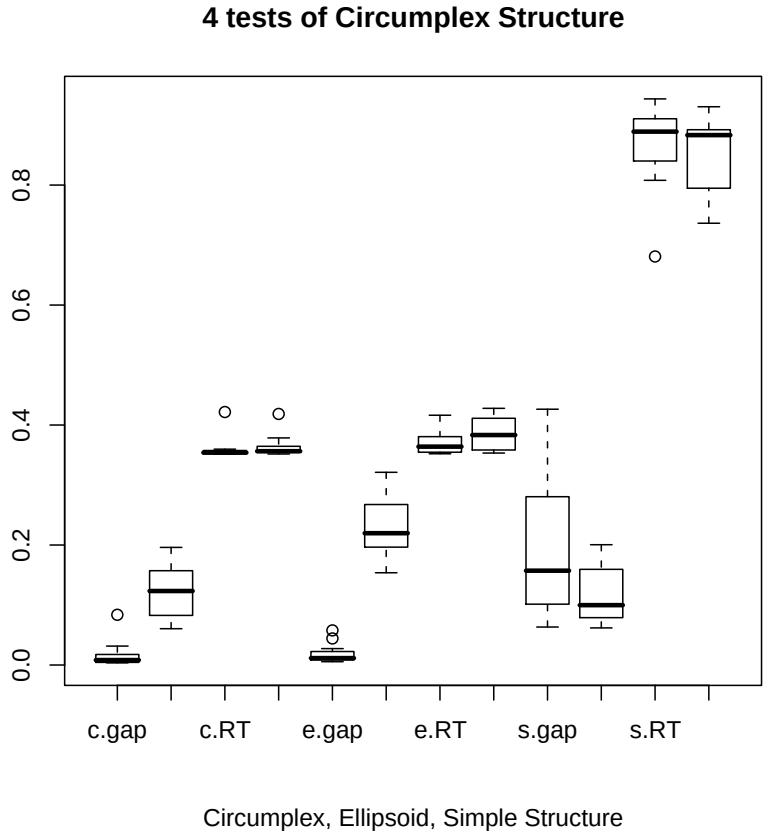

Figure 1: Tests for circumplex structure can be shown to discriminate circumplex and simple structure. Here we compare four tests of circumplex structure, the Gap Test, the Fisher Test, the Rotation Test, and the Variance Test of Circumplex structure. The data sets vary on whether or not they have a circumplex or simple structure. See Acton and Revelle (2004) for details.

Examples

```
circ.data <- circ.sim(24,500)
circ.fa <- factor.pa(circ.data,2)
#plot(circ.fa$loadings)
ct <- circ.tests(circ.fa)
#compare with non-circumplex data
simp.data <- item.sim(24,500)
simp.fa <- factor.pa(simp.data,2)
#plot(simp.fa$loadings)
st <- circ.tests(simp.fa)
print(rbind(ct,st),digits=2)
```

`cities` *Distances between 11 US cities*

Description

Airline distances between 11 US cities may be used as an example for multidimensional scaling or cluster analysis.

Usage

```
data(cities)
```

Format

A data frame with 11 observations on the following 11 variables.

`ATL` Atlana, Georgia

`BOS` Boston, Massachusetts

`ORD` Chicago, Illinois

`DCA` Washington, District of Columbia

`DEN` Denver, Colorado

`LAX` Los Angeles, California

`MIA` Miami, Florida

`JFK` New York, New York

`SEA` Seattle, Washington

`SFO` San Francisco, California

`MSY` New Orleans, Lousianna

Details

An 11 x11 matrix of distances between major US airports. This is a useful demonstration of multiple dimensional scaling.

city.location is a dataframe of longitude and latitude for those cities.

Note that the 2 dimensional MDS solution does not perfectly capture the data from these city distances. Boston, New York and Washington, D.C. are located slightly too far west, and Seattle and LA are slightly too far south.

Source

`http://www.timeanddate.com/worldclock/distance.html`

Examples

```
data(cities)
city.location[,1] <- -city.location[,1]
if(require(maps)) {map("usa")
title("MultiDimensional Scaling of US cities")
points(city.location)} else {plot(city.location, xlab="Dimension 1", ylab="Dimension 2",main ="multidimen
city.loc <- cmdscale(cities, k=2) #ask for a 2 dimensional solution  round(city.loc,0)
city.loc <- -city.loc
city.loc <- rescale(city.loc,mean(city.location),sd(city.location))
points(city.loc,type="n")
text(city.loc,labels=names(cities))
```

---

cluster.cor *Find correlations of composite variables from a larger matrix*

Description

Given a n x c cluster definition matrix of -1s, 0s, and 1s (the keys) , and a n x n correlation matrix, find the correlations of the composite clusters. The keys matrix can be entered by hand, copied from the clipboard (`read.clipboard`), or taken as output from the `factor2cluster` function.

Usage

```
cluster.cor(keys, r.mat, correct = TRUE,digits=2)
```

Arguments

`keys`	A matrix of cluster keys
`r.mat`	A correlation matrix
`correct`	TRUE shows both raw and corrected for attenuation correlations
`digits`	round off answer to digits

Details

This is one of the functions used in the SAPA procedures to form synthetic correlation matrices. Given any correlation matrix of items, it is easy to find the correlation matrix of scales made up of those items. This can also be done from the original data matrix using `score.items`.

A typical use in the SAPA project is to form item composites by clustering or factoring (see `factor.pa`, `ICLUST`, `principal`), extract the clusters from these results (`factor2cluster`), and then form the composite correlation matrix using `cluster.cor`. The variables in this reduced matrix may then be used in multiple correlatin procedures using `mat.regress`.

The original correlation is pre and post multiplied by the (transpose) of the keys matrix.

If some correlations are missing from the original matrix this will lead to missing values (NA) for scale intercorrelations based upon those lower level correlations.

Because the alpha estimate of reliability is based upon the correlations of the items rather than upon the covariances, this estimate of alpha is sometimes called "standardized alpha". If the raw items are available, it is useful to compare standardized alpha with the raw alpha found using `score.items`. They will differ substantially only if the items differ a great deal in their variances.

Value

`cor`	the (raw) correlation matrix of the clusters
`sd`	standard deviation of the cluster scores
`corrected`	raw correlations below the diagonal, alphas on diagonal, disattenuated above diagonal
`alpha`	The (standardized) alpha reliability of each scale.
`size`	How many items are in each cluster?

Note

See SAPA Revelle, W., Wilt, J., and Rosenthal, A. (2009) Personality and Cognition: The Personality-Cognition Link. In Gruszka, A. and Matthews, G. and Szymura, B. (Eds.) Handbook of Individual Differences in Cognition: Attention, Memory and Executive Control, Springer.

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu⟩

See Also

`factor2cluster`, `mat.regress`, `alpha.scale`, `score.items`

Examples

```
## Not run:
data(attitude)
keys <- matrix(c(1,1,1,0,0,0,0,
                 0,0,0,1,1,1,1),ncol=2)
colnames(keys) <- c("first","second")
r.mat <- cor(attitude)
cluster.cor(keys,r.mat)
## End(Not run)
#$cor
#       first second
#first    1.0    0.6
#second   0.6    1.0
#
#$sd
# first second
#  2.57   3.01
```

```
#
#$corrected
#       first second
#first   0.82   0.77
#second  0.60   0.74
#
#$size
# first second
#     3      4
```

`cluster.fit` *cluster Fit: fit of the cluster model to a correlation matrix*

Description

How well does the cluster model found by `ICLUST` fit the original correlation matrix? A similar algorithm `factor.fit` is found in `VSS`. This function is internal to ICLUST but has more general use as well.

In general, the cluster model is a Very Simple Structure model of complexity one. That is, every item is assumed to represent only one factor/cluster. Cluster fit is an analysis of how well this model reproduces a correlation matrix. Two measures of fit are given: cluster fit and factor fit. Cluster fit assumes that variables that define different clusters are orthogonal. Factor fit takes the loadings generated by a cluster model, finds the cluster loadings on all clusters, and measures the degree of fit of this somewhat more complicated model. Because the cluster loadings are similar to, but not identical to factor loadings, the factor fits found here and by `factor.fit` will be similar.

Usage

```
cluster.fit(original, load, clusters, diagonal = FALSE, digits = 2)
```

Arguments

`original`	The original correlation matrix being fit
`load`	Cluster loadings – that is, the correlation of individual items with the clusters, corrected for item overlap
`clusters`	The cluster structure
`diagonal`	Should we fit the diagonal as well?
`digits`	Number of digits to be used in the output

Details

The cluster model is similar to the factor model: R is fitted by C'C. Where C <- Cluster definition matrix x the loading matrix. How well does this model approximate the original correlation matrix and how does this compare to a factor model?

The fit statistic is a comparison of the original (squared) correlations to the residual correlations. Fit = 1 - r*2/r2 where r* is the residual correlation of data - model and model = C'C.

Value

clusterfit	The cluster model is a reduced form of the factor loading matrix. That is, it is the product of the elements of the cluster matrix * the loading matrix.
factorfit	How well does the complete loading matrix reproduce the correlation matrix?

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu⟩

References

http://personality-project.org/r/r.ICLUST.html

See Also

VSS, ICLUST, factor2cluster, cluster.cor, factor.fit

Examples

```
r.mat<- Harman74.cor$cov
iq.clus <- ICLUST(r.mat,nclusters =2)
fit <- cluster.fit(r.mat,iq.clus$loadings,iq.clus$clusters)
fit
```

---

cluster.loadings *Find item by cluster correlations, corrected for overlap and reliability*

Description

Given a n x n correlation matrix and a n x c matrix of -1,0,1 cluster weights for those n items on c clusters, find the correlation of each item with each cluster. If the item is part of the cluster, correct for item overlap. Part of the ICLUST set of functions, but useful for many item analysis problems.

Usage

```
cluster.loadings(keys, r.mat, correct = TRUE, digits = 2)
```

Arguments

keys	Cluster keys: a matrix of -1,0,1 cluster weights
r.mat	A correlation matrix
correct	Correct for reliability
digits	Number of digits output

Details

Given a set of items to be scored as (perhaps overlapping) clusters and the intercorrelation matrix of the items, find the clusters and then the correlations of each item with each cluster. Correct for item overlap by replacing the item variance with its average within cluster inter-item correlation.

Although part of ICLUST, this may be used in any SAPA application where we are interested in item- whole correlations of items and composite scales.

These loadings are particularly interpretable when sorted by absolute magnitude for each cluster (see `ICLUST.sort`).

Value

`loadings`	A matrix of item-cluster correlations (loadings)
`cor`	Correlation matrix of the clusters
`corrected`	Correlation matrix of the clusters, raw correlations below the diagonal, alpha on diagonal, corrected for reliability above the diagonal
`sd`	Cluster standard deviations
`alpha`	alpha reliabilities of the clusters
`count`	Number of items in the cluster

Note

Although part of ICLUST, this may be used in any SAPA application where we are interested in item- whole correlations of items and composite scales.

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu⟩

References

ICLUST: `http://personality-project.org/r/r.iclust.html`

See Also

`ICLUST`, `factor2cluster`, `cluster.cor`

Examples

```
## Not run:
 r.mat<- Harman74.cor$cov
 clusters <- matrix(c(1,1,1,rep(0,24),1,1,1,1,rep(0,17)),ncol=2)
 cluster.loadings(clusters,r.mat)
## End(Not run)
```

`cluster.plot` *Plot factor/cluster loadings and assign items to clusters by their highest loading.*

Description

Cluster analysis and factor analysis are procedures for grouping items in terms of a smaller number of (latent) factors or (observed) clusters. Graphical presentations of clusters typically show tree structures, although they can be represented in terms of item by cluster correlations.

Cluster.plot plots items by their cluster loadings (taken, e.g., from `ICLUST`) or factor loadings (taken, eg., from `factor.pa`). Cluster membership may be assigned apriori or may be determined in terms of the highest (absolute) cluster loading for each item.

If the input is an object of class "kmeans", then the cluster centers are plotted.

Usage

```
cluster.plot(ic.results, cluster = NULL, cut = 0, labels=NULL,title = "Cluster plot",...)
factor.plot(ic.results, cluster = NULL, cut = 0, labels=NULL,title = "Cluster plot",...)
```

Arguments

`ic.results`	A factor analysis or cluster analysis output including the loadings, or a matrix of item by cluster correlations. Or the output from a kmeans cluster analysis.
`cluster`	A vector of cluster membership
`cut`	Assign items to clusters if the absolute loadings are > cut
`labels`	If row.names exist they will be added to the plot, or, if they don't, labels can be specified. If labels =NULL, and there are no row names, then variables are labeled by row number.)
`title`	Any title
`...`	Further options to plot

Details

Results of either a factor analysis or cluster analysis are plotted. Each item is assigned to its highest loading factor, and then identified by variable name as well as cluster (by color).

Value

Graphical output is presented

Author(s)

William Revelle

See Also

ICLUST, ICLUST.graph, fa.graph

Examples

```
circ.data <- circ.sim(24,500)
circ.fa <- factor.pa(circ.data,2)
cluster.plot(circ.fa,cut=.5)
```

---

cluster2keys — *Convert a cluster vector (from e.g., kmeans) to a keys matrix suitable for scoring item clusters.*

Description

The output of the kmeans clustering function produces a vector of cluster membership. The score.items and cluster.cor functions require a matrix of keys. cluster2keys does this.

Usage

```
cluster2keys(c)
```

Arguments

c	A vector of cluster assignments or an object of class "kmeans" that contains a vector of clusters.

Details

Note that because kmeans will not reverse score items, the clusters defined by kmeans will not necessarily match those of ICLUST with the same number of clusters extracted.

Value

keys	A matrix of keys suitable for score.items or cluster.cor

Author(s)

William Revelle

See Also

cluster.cor,score.items

Examples

```
test.data <- Harman74.cor$cov
kc <- kmeans(test.data,4)
keys <- cluster2keys(kc)
keys  #these match those found by ICLUST
cluster.cor(keys,test.data)
```

`comorbidity` *Convert base rates of two diagnoses and their comorbidity into phi, Yule, and tetrachorics*

Description

In medicine and clinical psychology, diagnoses tend to be categorical (someone is depressed or not, someone has an anxiety disorder or not). Cooccurrence of both of these symptoms is called comorbidity. Diagnostic categories vary in their degree of comorbidity with other diagnostic categories. From the point of view of correlation, comorbidity is just a name applied to one cell in a four fold table. It is thus possible to analyze comorbidity rates by considering the probability of the separate diagnoses and the probability of the joint diagnosis. This gives the two by two table needed for a phi, Yule, or tetrachoric correlation.

Usage

```
comorbidity(d1, d2, com, labels = NULL)
```

Arguments

`d1`	Proportion of diagnostic category 1
`d2`	Proportion of diganostic category 2
`com`	Proportion of comorbidity (diagnostic category 1 and 2)
`labels`	Names of categories 1 and 2

Value

`twobytwo`	The two by two table implied by the input
`phi`	Phi coefficient of the two by two table
`Yule`	Yule coefficient of the two by two table
`tetra`	Tetrachoric coefficient of the two by two table

Note

Requires the polycor package

Author(s)

William Revelle

See Also

phi, Yule

Examples

```
if(require(polycor)) {comorbidity(.2,.15,.1,c("Anxiety","Depression")) }
```

`correct.cor` *Find dis-attenuated correlations given correlations and reliabilities*

Description

Given a raw correlation matrix and a vector of reliabilities, report the disattenuated correlations above the diagonal.

Usage

```
correct.cor(x, y)
```

Arguments

x	A raw correlation matrix
y	Vector of reliabilities

Details

Disattenuated correlations may be thought of as correlations between the latent variables measured by a set of observed variables. That is, what would the correlation be between two (unreliable) variables be if both variables were measured perfectly reliably.

This function is mainly used if importing correlations and reliabilities from somewhere else. If the raw data are available, use `score.items`, or `cluster.loadings` or `cluster.cor`.

Examples of the output of this function are seen in `cluster.loadings` and `cluster.cor`

Value

Raw correlations below the diagonal, reliabilities on the diagonal, disattenuated above the diagonal.

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu⟩

References

http://personality-project.org/revelle/syllabi/405.syllabus.html

See Also

cluster.loadings and cluster.cor

Examples

```
# attitude from the datasets package
#example 1 is a rather clunky way of doing things

a1 <- attitude[,c(1:3)]
a2 <- attitude[,c(4:7)]
x1 <- rowSums(a1)  #find the sum of the first 3 attitudes
x2 <- rowSums(a2)   #find the sum of the last 4 attitudes
alpha1 <- alpha.scale(x1,a1)
alpha2 <- alpha.scale(x2,a2)
x <- matrix(c(x1,x2),ncol=2)
x.cor <- cor(x)
alpha <- c(alpha1,alpha2)
round(correct.cor(x.cor,alpha),2)
#
#much better - although uses standardized alpha
clusters <- matrix(c(rep(1,3),rep(0,7),rep(1,4)),ncol=2)
cluster.loadings(clusters,cor(attitude))
# or
clusters <- matrix(c(rep(1,3),rep(0,7),rep(1,4)),ncol=2)
cluster.cor(clusters,cor(attitude))
#
#best
scores <- score.items(matrix(c(rep(1,3),rep(0,7),rep(1,4)),ncol=2),attitude)
scores$corrected
```

---

cortest.bartlett *Bartlett's test that a correlation matrix is an identity matrix*

Description

Bartlett (1951) proposed that -ln(det(R)*(N-1 - (2p+5)/6) was distributed as chi square if R were an identity matrix. A useful test that residuals correlations are all zero.

Usage

```
cortest.bartlett(R, n = NULL)
```

Arguments

R	A correlation matrix. (If R is not square, correlations are found and a warning is issued.
n	Sample size (if not specified, 100 is assumed.

Details

More useful for pedagogical purposes than actual applications. The Bartlett test is asymptotically chi square distributed.

Value

`chisq`	Assymptotically chisquare
`p.value`	Of chi square
`df`	The degrees of freedom

Author(s)

William Revelle

References

Bartlett, M. S., (1951), The Effect of Standardization on a chi square Approximation in Factor Analysis, Biometrika, 38, 337-344.

See Also

`cortest.mat`, `cortest.normal`, `cortest.jennrich`

Examples

```
set.seed(42)
x <- matrix(rnorm(1000),ncol=10)
r <- cor(x)
cortest.bartlett(r)      #random data don't differ from an identity matrix
data(bfi)
cortest.bartlett(bfi)    #not an identity matrix
```

`cortest.mat` *Chi square tests of whether a single matrix is an identity matrix, or a pair of matrices are equal.*

Description

Steiger (1980) pointed out that the sum of the squared elements of a correlation matrix, or the Fisher z score equivalents, is distributed as chi square under the null hypothesis that the values are zero (i.e., elements of the identity matrix). This is particularly useful for examining whether correlations in a single matrix differ from zero or for comparing two matrices. Jennrich (1970) also examined tests of differences between matrices.

Usage

```
cortest.normal(R1, R2 = NULL, n1 = NULL, n2 = NULL, fisher = TRUE)
cortest(R1,R2=NULL,n1=NULL,n2 = NULL, fisher = TRUE)   #same as cortest.normal
cortest.mat(R1,R2=NULL,n1=NULL,n2 = NULL)
cortest.jennrich(R1,R2,n1=NULL, n2=NULL)
```

Arguments

R1	A correlation matrix. (If R1 is not rectangular, the correlations are found).
R2	A correlation matrix. If R2 is not rectangular, the correlations are found. If R2 is NULL, then the test is just whether R1 is an identity matrix.
n1	Sample size of R1
n2	Sample size of R2
fisher	Fisher z transform the correlations?

Details

There are several ways to test if a matrix is the identity matrix. The most well known is the chi square test of Bartlett (1951) and Box (1949). A very straightforward test, discussed by Steiger (1980) is to find the sum of the squared correlations or the sum of the squared Fisher transformed correlations. Under the null hypothesis that all the correlations are equal, this sum is distributed as chi square.

Yet another test, is the Jennrich(1970) test of the equality of two matrices.

Value

chi2	The chi square statistic
df	Degrees of freedom for the Chi Square
prob	The probability of observing the Chi Square under the null hypothesis.

Note

Both the cortest.jennrich and cortest.normal are probably overly stringent. The ChiSquare values for pairs of random samples from the same population are larger than would be expected. This is a good test for rejecting the null of no differences.

Author(s)

William Revelle

References

Steiger, James H. (1980) Testing pattern hypotheses on correlation matrices: alternative statistics and some empirical results. Multivariate Behavioral Research, 15, 335-352.

See Also

cortest.bartlett

Examples

```
x <- matrix(rnorm(1000),ncol=10)
y <- matrix(rnorm(500),ncol=10)
cortest.normal(x)  #just test if this matrix is an identity
cortest.normal(x,y) #do these two matrices differ?
cortest.mat(x)
cortest.mat(x,y)   #twice the degrees of freedom as the Jennrich
cortest.jennrich(x,y) #
```

---

cosinor *Functions for analysis of circadian or diurnal data*

Description

Circadian data are periodic with a phase of 24 hours. These functions find the best fitting phase angle (cosinor), the circular mean, circular correlation with circadian data, and the linear by circular correlation

Usage

```
cosinor(data, code = NULL, period = 24)
circadian.mean(angle, hours=TRUE)
circadian.cor(angle, hours=TRUE)
circadian.linear.cor(angle,x,hours=TRUE)
```

Arguments

data	A data frame or matrix of observed values with the time of day as the first value (unless specified in code
code	Time of day of measurements
period	Although time of day is assumed to have a 24 hour rhythm, other rhythms may be fit.
angle	A vector, matrix, or data frame of phase angles (either as hours or as radians).
hours	If TRUE, measures are in 24 hours to the day, otherwise, radians
x	A set of external variables to correlate with the phase angles

Details

When data represent angles (such as the hours of peak alertness or peak tension during the day), we need to apply circular statistics rather than the more normal linear statistics (see Jammalamadaka(2006) for a very clear set of examples of circular statistics). The generalization of the mean to circular data is to convert each angle into a vector, average the x and y coordinates, and convert the result back to an angle. The gneralization of Pearson correlation to circular statistics is straight forward and is implemented in cor.circular in the circular package and in circadian.cor here. Just as the Pearson r is a ratio of covariance to the square root of the product of two variances, so is the circular correlation. The circular covariance of two circular vectors is defined as the average product of the sines of the deviations from the circular mean. The variance is thus the average squared sine of the angular deviations from the circular mean. Circular statistics are used for data that vary over a period (e.g., one day) or over directions (e.g., wind direction or bird flight). Jammalamadaka and Lund (2006) givesa very good example of the use of circular statistics in calculating wind speed and direction. The code from CircStats and circular was adapted to allow for an analysis of data from an experimental study of mood over the day.

Value

Describe the value returned If it is a LIST, use

`phase`	The phase angle that best fits the data
`fit`	Value of the correlation of the fit
`mean.angle`	A vector of mean angles
`R`	A matrix of circular correlations or linear by circular correlations

Author(s)

William Revelle

References

See circular statistics Jammalamadaka, Sreenivasa and Lund, Ulric (2006),The effect of wind direction on ozone levels: a case study ,Environmental and Ecological Statistics, 13, 287-298.

See Also

See the circular and CircStats packages.

`count.pairwise` *Count number of pairwise cases for a data set with missing (NA) data.*

Description

When doing cor(x, use= "pairwise"), it is nice to know the number of cases for each pairwise correlation. This is particularly useful when doing SAPA type analyses.

Usage

```
count.pairwise(x, y = NULL)
```

Arguments

`x`	An input matrix, typically a data matrix ready to be correlated.
`y`	An optional second input matrix

Value

result = matrix of counts of pairwise observations

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu⟩

Examples

```
## Not run:
x <- matrix(rnorm(1000),ncol=6)
y <- matrix(rnorm(500),ncol=3)
x[x < 0] <- NA
y[y> 1] <- NA

count.pairwise(x)
count.pairwise(y)
count.pairwise(x,y)
## End(Not run)
```

`cubits` *Galton's example of the relationship between height and 'cubit' or forearm length*

Description

Francis Galton introduced the 'co-relation' in 1888 with a paper discussing how to measure the relationship between two variables. His primary example was the relationship between height and forearm length. The data table (cubits) is taken from Galton (1888). Unfortunately, there seem to be some errors in the original data table in that the marginal totals do not match the table.

The data frame, `heights`, is converted from this table.

Usage

```
data(cubits)
```

Format

A data frame with 9 observations on the following 8 variables.

`16.5` Cubit length of lowest category

`16.75` a numeric vector

`17.25` a numeric vector

`17.75` a numeric vector

`18.25` a numeric vector

`18.75` a numeric vector

`19.25` a numeric vector

`19.75` a numeric vector

Details

Sir Francis Galton (1888) published the first demonstration of the correlation coefficient. The regression (or reversion to mediocrity) of the height to the length of the left forearm (a cubit) was found to .8. There seem to be some errors in the table as published in that the row sums do not agree with the actual row sums. These data are used to create a matrix using `table2matrix` for demonstrations of analysis and displays of the data.

Source

Galton (1888)

References

Galton, Francis (1888) Co-relations and their measurement. Proceedings of the Royal Society. London Series,45,135-145,

See Also

table2matrix, table2df,heights,ellipses,galton

Examples

```
data(cubits)
cubits
heights <- table2df(cubits,labs <- c("height","cubit"))
ellipses(heights,n=1,main="Galton's co-relation data set")
ellipses(jitter(heights$cubit,3),jitter(heights$height,3),pch=".",main="Galton's co-relation data set") #a
```

---

describe.by *Basic summary statistics by group*

Description

Report basic summary statistics by a grouping variable. Useful if the grouping variable is some experimental variable and data are to be aggregated for plotting. Just a wrapper for by and describe.

Usage

```
describe.by(x, group,mat=FALSE,...)
```

Arguments

x	a data.frame or matrix
group	a grouping variable or a list of grouping variables
mat	provide a matrix output rather than a list
...	parameters to be passed to describe

Details

To get descriptive statistics for several different grouping variables, make sure that group is a list!

Value

A data.frame of the relevant statistics broken down by group:
item name
item number
number of valid cases
mean
standard deviation
median
mad: median absolute deviation (from the median)

minimum
maximum
skew
standard error

Author(s)

William Revelle

See Also

`describe`

Examples

```
data(sat.act)
describe.by(sat.act,sat.act$gender) #just one grouping variable
describe.by(sat.act,list(sat.act$gender,sat.act$education))  #two grouping variables
des.mat <- describe.by(sat.act$age,sat.act$education,mat=TRUE) #matrix output
des.mat <- describe.by(sat.act$age,list(sat.act$education,sat.act$gender),mat=TRUE)
```

`describe` *Basic descriptive statistics useful for psychometrics*

Description

There are many summary statistics available in R; this function provides the ones most useful for scale construction and item analysis in classic psychometrics. Range is most useful for the first pass in a data set, to check for coding errors.

Usage

```
describe(x, digits = 2, na.rm = TRUE, interp=FALSE,skew = TRUE, ranges = TRUE,trim=.1)
```

Arguments

`x`	A data frame or matrix
`digits`	How many significant digits to report
`na.rm`	The default is to delete missing data. na.rm=FALSE will delete the case.
`interp`	Should the median be standard or interpolated
`skew`	Should the skew and kurtosis be calculated?
`ranges`	Should the range be calculated?
`trim`	trim=.1 – trim means by dropping the top and bottom trim fraction

Details

In basic data analysis it is vital to get basic descriptive statistics. Procedures such as `summary` and hmisc::describe do so. The describe function in the `psych` package is meant to produce the most frequently requested stats in psychometric and psychology studies, and to produce them in an easy to read data.frame. The results from describe can be used in graphics functions (e.g., `error.crosses`).

The range statistics (min, max, range) are most useful for data checking to detect coding errors, and should be found in early analyses of the data.

Although describe will work on data frames as well as matrices, it is important to realize that for data frames, descriptive statistics will be reported only for those variables where this makes sense (i.e., not for factors or for alphanumeric data).

In a typical study, one might read the data in from the clipboard (`read.clipboard`), show the splom plot of the correlations (`pairs.panels`), and then describe the data.

na.rm=FALSE is equivalent to describe(na.omit(x))

Value

A data.frame of the relevant statistics:
item name
item number
number of valid cases
mean
standard deviation
trimmed mean (with trim defaulting to .1)
median (standard or interpolated
mad: median absolute deviation (from the median)
minimum
maximum
skew
kurtosis
standard error

Note

Describe uses either the mean or colMeans functions depending upon whether the data are a data.frame or a matrix. The mean function supplies means for the columns of a data.frame, but the overall mean for a matrix. Mean will throw a warning for non-numeric data, but colMeans stops with non-numeric data. Thus, the describe function uses either mean (for data frames) or colMeans (for matrices). This is true for skew and kurtosi as well.

Author(s)

`http://personality-project.org/revelle.html`

Maintainer: William Revelle ⟨revelle@northwestern.edu⟩

See Also

describe.by, skew, kurtosi interp.median, pairs.panels, read.clipboard, error.crosses

Examples

```
describe(attitude)

describe(attitude,skew=FALSE)   #attitude is taken from R data sets
```

---

eigen.loadings *Convert eigen vectors and eigen values to the more normal (for psychologists) component loadings*

Description

The default procedures for principal component returns values not immediately equivalent to the loadings from a factor analysis. eigen.loadings translates them into the more typical metric of eigen vectors multiplied by the squareroot of the eigenvalues. This lets us find pseudo factor loadings if we have used princomp or eigen.
If we use principal to do our principal components analysis, then we do not need this routine.

Usage

```
eigen.loadings(x)
```

Arguments

x	the output from eigen or a list of class princomp derived from princomp

Value

A matrix of Principal Component loadings more typical for what is expected in psychometrics. That is, they are scaled by the square root of the eigenvalues.

Note

Useful for SAPA analyses

Author(s)

⟨ revelle@northwestern.edu ⟩
http://personality-project.org/revelle.html

Examples

```
x <- eigen(Harman74.cor$cov)
x$vectors[1:8,1:4]  #as they appear from eigen
y <- princomp(covmat=Harman74.cor$cov)
y$loadings[1:8,1:4] #as they appear from princomp
eigen.loadings(x)[1:8,1:4] # rescaled by the eigen values
```

---

ellipses *Plot data and 1 and 2 sigma correlation ellipses*

Description

For teaching correlation, it is useful to draw ellipses around the mean to reflect the correlation. This variation of the ellipse function from John Fox's car package does so. Input may be either two vectors or a matrix or data.frame. In the latter cases, if the number of variables >2, then the ellipses are done in the `pairs.panels` function. Ellipses may be added to existing plots.

Usage

```
ellipses(x, y = NULL, add = FALSE, smooth=TRUE, lm=FALSE,data=TRUE, n = 2,span=2/3, iter=3, col
```

Arguments

`x`	a vector,matrix, or data.frame
`y`	Optional second vector
`add`	Should a new plot be created, or should it be added to?
`smooth`	smooth = TRUE -> draw a loess fit
`lm`	lm=TRUE -> draw the linear fit
`data`	data=TRUE implies draw the data points
`n`	Should 1 or 2 ellipses be drawn
`span`	averaging window parameter for the lowess fit
`iter`	iteration parameter for lowess
`col`	color of ellipses (default is red
`xlab`	label for the x axis
`ylab`	label for the y axis
`...`	Other parameters for plotting

Details

Ellipse dimensions are calculated from the correlation between the x and y variables and are scaled as sqrt(1+r) and sqrt(1-r).

Value

A single plot (for 2 vectors or data frames with fewer than 3 variables. Otherwise a call is made to `pairs.panels`.

Note

Adapted from John Fox's ellipse and data.ellipse functions.

Author(s)

William Revelle

References

Galton, Francis (1888), Co-relations and their measurement. Proceedings of the Royal Society. London Series, 45, 135-145.

See Also

`pairs.panels`

Examples

```
data(galton)
ellipses(galton,lm=TRUE)
ellipses(galton$parent,galton$child,xlab="Mid Parent Height",ylab="Child Height") #input are two vectors
data(sat.act)
ellipses(sat.act)  #shows the pairs.panels ellipses
```

---

epi.bfi *13 personality scales from the Eysenck Personality Inventory and Big 5 inventory*

Description

A small data set of 5 scales from the Eysenck Personality Inventory, 5 from a Big 5 inventory, a Beck Depression Inventory, and State and Trait Anxiety measures. Used for demonstrations of correlations, regressions, graphic displays.

Usage

```
data(epi.bfi)
```

Format

A data frame with 231 observations on the following 13 variables.

`epiE` EPI Extraversion

`epiS` EPI Sociability (a subset of Extraversion items

`epiImp` EPI Impulsivity (a subset of Extraversion items

`epilie` EPI Lie scale

`epiNeur` EPI neuroticism

`bfagree` Big 5 inventory (from the IPIP) measure of Agreeableness

`bfcon` Big 5 Conscientiousness

`bfext` Big 5 Extraversion

`bfneur` Big 5 Neuroticism

`bfopen` Big 5 Openness

`bdi` Beck Depression scale

`traitanx` Trait Anxiety

`stateanx` State Anxiety

Details

Self report personality scales tend to measure the "Giant 2" of Extraversion and Neuroticism or the "Big 5" of Extraversion, Neuroticism, Agreeableness, Conscientiousness, and Openness. Here is a small data set from Northwestern University undergraduates with scores on the Eysenck Personality Inventory (EPI) and a Big 5 inventory taken from the International Personality Item Pool.

Source

Data were collected at the Personality, Motivation, and Cognition Lab (PMCLab) at Northwestern by William Revelle)

References

`http://personality-project.org/pmc.html`

Examples

```
data(epi.bfi)
pairs.panels(epi.bfi[,1:5])
describe(epi.bfi)
```

`error.bars.by` *Plot means and confidence intervals for multiple groups*

Description

One of the many functions in R to plot means and confidence intervals. Meant mainly for demonstration purposes for showing the probabilty of replication from multiple samples. Can also be combined with such functions as boxplot to summarize distributions. Means and standard errors for each group are calculated using `describe.by`.

Usage

```
error.bars.by(x, group,by.var = FALSE,x.cat=TRUE, ylab = "NULL", xlab = "NULL",main=NULL, ylim
```

Arguments

`x`	A data frame or matrix
`group`	A grouping variable
`by.var`	A different line for each group (default) or each variable
`x.cat`	Is the grouping variable categorical (TRUE) or continuous (FALSE
`ylab`	y label
`xlab`	x label
`main`	title for figure
`ylim`	if specified, the limits for the plot, otherwise based upon the data
`alpha`	alpha level of confidence interval. Default is 1- alpha =95% confidence interval
`labels`	X axis label
`pos`	where to place text: below, left, above, right
`arrow.len`	How long should the top of the error bars be?
`add`	add=FALSE, new plot, add=TRUE, just points and error bars
`...`	other parameters to pass to the plot function, e.g., typ="b" to draw lines, lty="dashed" to draw dashed lines

Details

Drawing the mean +/- a confidence interval is a frequently used function when reporting experimental results. By default, the confidence interval is 1.96 standard errors.

This function is a wrapper for `error.bars` and allows groups to be organized either as the x axis or as separate lines.

Value

Graphic output showing the means + x% confidence intervals for each group. For ci=1.96, and normal data, this will be the 95% confidence region. For ci=1, the 68% confidence region.

See Also

See Also as `error.crosses`, `error.bars`

Examples

```
x <- matrix(rnorm(500),ncol=20)
y <- sample(4,25 ,replace=TRUE)
x <- x+y
error.bars.by(x,y)
error.bars.by(x,y,TRUE)
```

`error.bars` *Plot means and confidence intervals*

Description

One of the many functions in R to plot means and confidence intervals. Meant mainly for demonstration purposes for showing the probabilty of replication from multiple samples. Can also be combined with such functions as boxplot to summarize distributions. Means and standard errors are calculated from the raw data using `describe`.

Usage

```
error.bars(x, ylab = "Dependent Variable",xlab="Independent Variable", main=NULL, ylim = NULL,
```

Arguments

`x`	A data frame or matrix
`ylab`	y label
`xlab`	x label
`main`	title for figure
`ylim`	if specified, the limits for the plot, otherwise based upon the data
`alpha`	alpha level of confidence interval – defaults to 95% confidence interval
`labels`	X axis label
`pos`	where to place text: below, left, above, right
`arrow.len`	How long should the top of the error bars be?
`add`	add=FALSE, new plot, add=TRUE, just points and error bars
`...`	other parameters to pass to the plot function, e.g., typ="b" to draw lines, lty="dashed" to draw dashed lines

Details

Drawing the mean +/- a confidence interval is a frequently used function when reporting experimental results. By default, the confidence interval is 1.96 standard errors.

Value

Graphic output showing the means + x

Author(s)

William Revelle

See Also

`error.crosses` for two way error bars, `error.bars.by` for error bars for different groups

Examples

```
x <- matrix(rnorm(500),ncol=20)
error.bars(x)
#now do a boxplot and then add error bars
x.df <- as.data.frame(x)
boxplot(x.df)
error.bars(x.df, add=TRUE)

error.bars(attitude) #another example
```

`error.crosses` *Plot x and y error bars*

Description

Given two vectors of data, plot the means and show standard errors in both X and Y directions.

Usage

```
error.crosses(x, y, labels = NULL, pos = NULL, arrow.len = 0.2, ...)
```

Arguments

`x`	A vector of summary statistics (from Describe)
`y`	A second vector of summary statistics (also from Describe)
`labels`	name the pair
`pos`	Labels are located where with respect to the mean?
`arrow.len`	Arrow length
`...`	Other parameters for plot

Details

For an example of two way error bars describing the effects of mood manipulations upon positive and negative affect, see `http://personality-project.org/revelle/publications/happy-sad-appendix/FIG.A-6.pdf`

The second example shows how error crosses can be done for multiple variables where the grouping variable is found dynamically.

Author(s)

William Revelle
⟨revelle@northwestern.edu⟩

See Also

To draw error bars for single variables `error.bars`, or by groups `error.bars.by`, or to find descriptive statistics `describe` or descriptive statistics by a grouping variable `describe.by`

Examples

```
desc <- describe(attitude)
x <- desc[1,]
y <- desc[2,]
plot(x$mean,y$mean,xlab=rownames(x),ylab=rownames(y))   #in graphics window
error.crosses(x,y)    #in graphics window
#now for a bit more complicated plotting
desc <- describe.by(attitude,(attitude[,7]>41)) #select a high and low group
g1 <- desc$'FALSE'
g2 <- desc$'TRUE'
plot(g1$mean,g2$mean,xlab = "Low Advance",ylab="High Advance",xlim=c(30,80),ylim=c(50,80))
error.crosses(g1,g2,labels=rownames(g1),pos=rep(1,7))
title("Attitudes grouped by high and low scores on Advance")
```

`fa.graph` *Graph factor loading matrices*

Description

Factor analysis or principal components analysis results are typically interpreted in terms of the major loadings on each factor. These structures may be represented as a table of loadings or graphically, where all loadings with an absolute value $>$ some cut point are represented as an edge (path).

Usage

```
fa.graph(fa.results, out.file = NULL, labels = NULL, cut = 0.3, simple = TRUE, size = c(8, 6),
```

```
A graphNEL graph with directed edges
Number of Nodes = 5
Number of Edges = 4
```

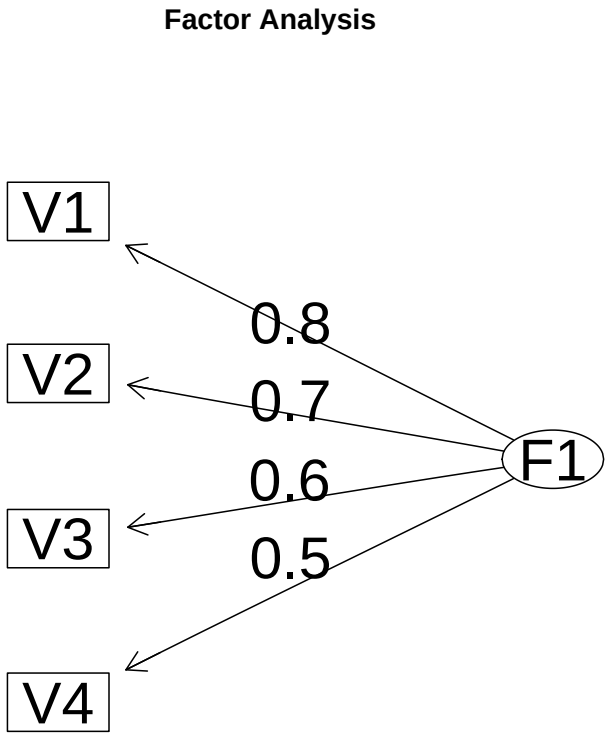

Figure 2: Congeneric tests have one common factor and can differ in their error variances. A demonstration of congeneric.sim, factor.pa, and fa.graph.

Arguments

`fa.results`	The output of factor analysis or principal components analysis
`out.file`	If it exists, a dot representation of the graph will be stored here
`labels`	Variable labels
`cut`	Loadings with abs(loading) > cut will be shown
`simple`	Only the biggest loading per item is shown
`size`	graph size
`node.font`	
`edge.font`	
`rank.direction`	
`digits`	Number of digits to show as an edgelable
`main`	Graphic title
`...`	other parameters

Details

Path diagram representations have become standard in confirmatory factor analysis, but are not yet common in exploratory factor analysis. Representing factor structures graphically helps some people understand the structure.

Although a nice graph is drawn for the orthogonal factor case, the oblique factor drawing is acceptable, but is better if cleaned up outside of R.

Value

A graph is drawn using rgraphviz. If an output file is specified, the graph instructions are also saved in the dot language.

Note

Requires Rgraphviz. For the Mac, there are occasional difficulties installing Rgraphviz from Bioconductor in that some libraries are misplaced and need to be relinked using X11.

As of June 1, 2007 there is an occasionally strange result when using the simple=FALSE option in Sweave.

Author(s)

William Revelle

See Also

`omega.graph`, `ICLUST.graph`

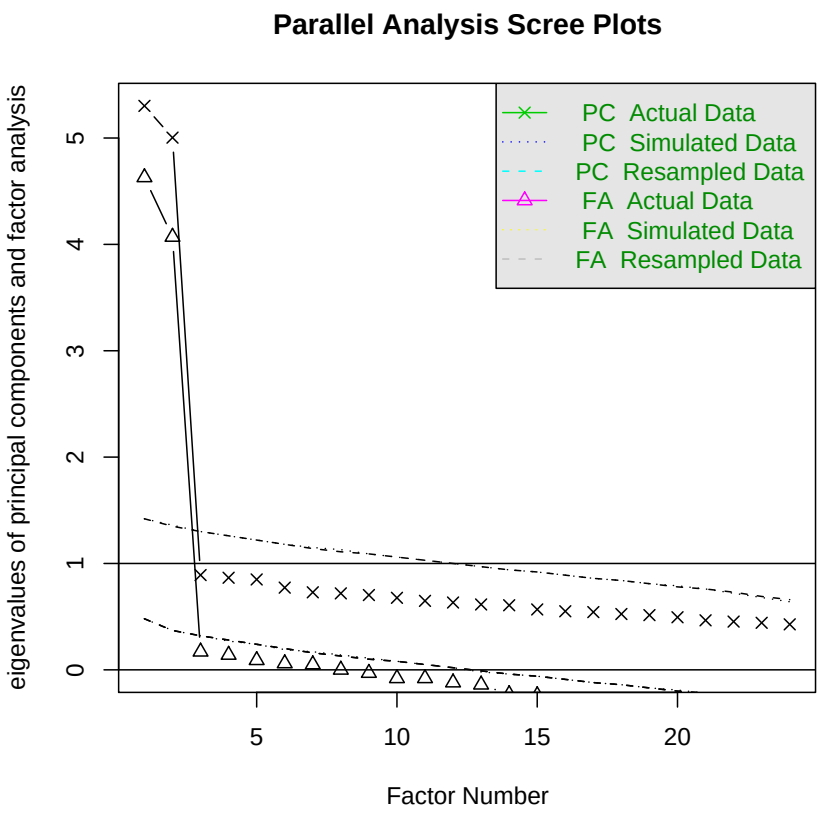

Figure 3: Plots of eigen values for real and simulated "parallel" data give two ways to determine the optimal number of factors: Examine the scree for large breaks or find when the eigen values of random data are greater than for the real data. For this data set, both methods suggest two factors are optimal.

Examples

```
test.simple <- factor.pa(item.sim(16),2)
if(require(Rgraphviz)) {fa.graph(test.simple) }
```

`fa.parallel` *Scree plots of data or correlation matrix compared to random "parallel" matrices*

Description

One way to determine the number of factors or components in a data matrix or a correlation matrix is to examine the "scree" plot of the successive eigenvalues. Sharp breaks in the plot suggest the appropriate number of components or factors to extract. "Parallel" analyis is an alternative technique that compares the scree of the observed data with that of a random data matrix of the same size as the original.

Usage

```
fa.parallel(x, n.obs = NULL, fa="both", main = "Parallel Analysis Scree Plots",ntrials=20,error
```

Arguments

`x`	A data.frame or data matrix of scores. If the matrix is square, it is assumed to be a correlation matrix. Otherwise, correlations (with pairwise deletion) will be found
`n.obs`	n.obs=0 implies a data matrix/data.frame. Otherwise, how many cases were used to find the correlations.
`fa`	show the eigen values for a principal components (fa="pc") or a principal axis factor analysis (fa="fa") or both principal components and principal factors (fa="both")
`main`	a title for the analysis
`ntrials`	Number of simulated analyses to perform
`error.bars`	Should error.bars be plotted (default = FALSE)

Details

Cattell's "scree" test is one of most simple tests for the number of factors problem. Humphreys and Montanelli's "parallel" analysis is an equally compelling procedure. Other procedures for determining the most optimal number of factors include finding the Very Simple Structure (VSS) criterion (`VSS`) and Velicer's `MAP` procedure (included in `VSS`). fa.parallel plots the eigen values for a principal components and principal factor solution and does the same for random matrices of the same size as the original data matrix. For raw data, the random matrices are 1) a matrix of univariate normal data and 2) random samples (randomized across rows) of the original data.

The means of (n.trials) random solutions are shown. Error bars are usually very small and are suppressed by default but can be shown if requested.

Value

A plot of the eigen values for the original data, ntrials of resampling of the original data, and of a equivalent size matrix of random normal deviates. If the data are a correlation matrix, specify the number of observations.

Author(s)

William Revelle

References

Humphreys, Lloyd G. and Montanelli, Richard G. (1975),An investigation of the parallel analysis criterion for determining the number of common factors. Multivariate Behavioral Research, 10, 193-205.

See Also

VSS,VSS.plot, VSS.parallel

Examples

```
test.data <- Harman74.cor$cov
fa.parallel(test.data,n.obs=200)

fa.parallel(attitude)
#
```

---

factor.congruence *Coefficient of factor congruence*

Description

Given two sets of factor loadings, report their degree of congruence (vector cosine).

Usage

```
factor.congruence(x, y,digits=2)
```

Arguments

x	A matrix of factor loadings
y	A second matrix of factor loadings
digits	Round off to digits

Details

Find the coefficient of factor congruence between two sets of factor loadings.

Factor congruences are the cosines of pairs vectors defined by the loadings matrix and based at the origin. Thus, for loadings that differ only by a scaler (e.g. the size of the eigen value), the factor congruences will be 1.

It is an interesting exercise to compare factor congruences with the correlations of factor loadings. Factor congruences are based upon the raw cross products, while correlations are based upon centered cross products. That is, correlations of factor loadings are cosines of the vectors based at the mean loading for each factor.

Input may either be matrices or factor analysis or principal components analyis output (which includes a loadings object), or a mixture of the two.

Value

A matrix of factor congruences.

Author(s)

⟨revelle@northwestern.edu⟩
`http://personality-project.org/revelle.html`

References

Gorsuch, Richard, (1983) Factor Analysis. Lawrence Erlebaum Associates.
Revelle, W. (In preparation) An Introduction to Psychometric Theory with applications in R (`http://personality-project.org/r/book/`)

See Also

`principal`, `factor.pa`

Examples

```
#fa <- factanal(x,4,covmat=Harman74.cor$cov)
#pc <- principal(Harman74.cor$cov,4)
#pa <- factor.pa(Harman74.cov$cor,4)
#factor.congruence(fa,pc)
#
#    Factor1 Factor2 Factor3 Factor4
#PC1    1.00    0.60    0.45    0.55
#PC2    0.44    0.49    1.00    0.56
#PC3    0.54    0.99    0.44    0.55
#PC4    0.47    0.52    0.48    0.99

# pa <- factor.pa(Harman74.cor$cov,4)
# factor.congruence(fa,pa)
#         PA1  PA3  PA2  PA4
#Factor1 1.00 0.61 0.46 0.55
#Factor2 0.61 1.00 0.50 0.60
#Factor3 0.46 0.50 1.00 0.57
#Factor4 0.56 0.62 0.58 1.00

#compare with
#round(cor(fa$loading,pc$loading),2)
```

`factor.fit` *How well does the factor model fit a correlation matrix. Part of the VSS package*

Description

The basic factor or principal components model is that a correlation or covariance matrix may be reproduced by the product of a factor loading matrix times its transpose: F'F or P'P. One simple index of fit is the 1 - sum squared residuals/sum squared original correlations. This fit index is used by `VSS`, `ICLUST`, etc.

Usage

```
factor.fit(r, f)
```

Arguments

`r`	a correlation matrix
`f`	A factor matrix of loadings.

Details

There are probably as many fit indices as there are psychometricians. This fit is a plausible estimate of the amount of reduction in a correlation matrix given a factor model. Note that it is sensitive to the size of the original correlations. That is, if the residuals are small but the original correlations are small, that is a bad fit.

Value

fit

Author(s)

William Revelle

See Also

`VSS`, `ICLUST`

Examples

```
## Not run:
#compare the fit of 4 to 3 factors for the Harman 24 variables
fa4 <- factanal(x,4,covmat=Harman74.cor$cov)
round(factor.fit(Harman74.cor$cov,fa4$loading),2)
#[1] 0.9
fa3 <- factanal(x,3,covmat=Harman74.cor$cov)
round(factor.fit(Harman74.cor$cov,fa3$loading),2)
#[1] 0.88

## End(Not run)
```

`factor.model` *Find R = F F' + U2 is the basic factor model*

Description

The basic factor or principal components model is that a correlation or covariance matrix may be reproduced by the product of a factor loading matrix times its transpose. Find this reproduced matrix. Used by `factor.fit`, `VSS`, `ICLUST`, etc.

Usage

```
factor.model(f)
```

Arguments

`f`	A matrix of loadings.

Value

A correlation or covariance matrix.

Author(s)

⟨revelle@northwestern.edu ⟩
`http://personality-project.org/revelle.html`

References

Gorsuch, Richard, (1983) Factor Analysis. Lawrence Erlebaum Associates.
Revelle, W. In preparation) An Introduction to Psychometric Theory with applications in R (`http://personality-project.org/r/book/`)

See Also

`ICLUST.graph`,`ICLUST.cluster`, `cluster.fit` , `VSS`, `omega`

Examples

```
f2 <- matrix(c(.9,.8,.7,rep(0,6),.6,.7,.8),ncol=2)
mod <- factor.model(f2)
round(mod,2)
```

```
A graphNEL graph with directed edges
Number of Nodes = 28
Number of Edges = 35
```

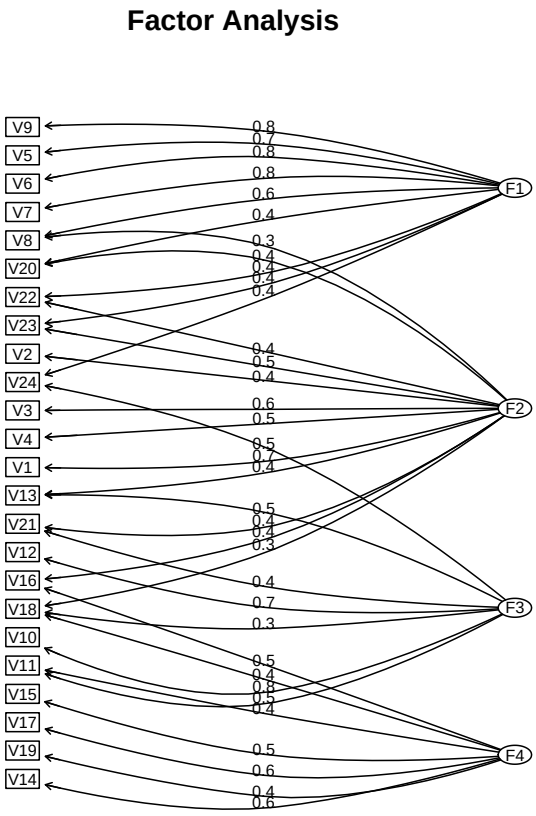

Figure 4: Four factors of the Harman 24 mental tests suggest that a simple structure is difficult to obtain. Note the cross loadings for several items. This figure was created with factor.pa and omega.graph. Compare with a hierarchical solution using the omega function (figure 6) or a cluster solution with ICLUST (figure 5).

`factor.pa` *Principal Axis Factor Analysis*

Description

Among the many ways to do factor analysis, one of the most conventional is principal axes. An eigen value decomposition of a correlation matrix is done and then the communalities for each variable are estimated by the first n factors. These communalities are entered onto the diagonal and the procedure is repeated until the sum(diag(r)) does not vary. For well behaved matrices, maximum likelihood factor analysis (factanal) is probably preferred.

Usage

```
factor.pa(r, nfactors=1, residuals = FALSE, rotate = "varimax",n.obs = NA,
scores = FALSE,SMC=TRUE, missing=FALSE,impute="median",min.err = 0.001, digits = 2, max.iter =
```

Arguments

`r`	A correlation matrix or a raw data matrix. If raw data, the correlation matrix will be found using pairwise deletion.
`nfactors`	Number of factors to extract, default is 1
`residuals`	Should the residual matrix be shown
`rotate`	"none", "varimax", "promax" or "oblimin" are possible rotations of the solution.
`n.obs`	Number of observations used to find the correlation matrix if using a correlation matrix. Used for finding the goodness of fit statistics.
`scores`	If TRUE, estimate factor scores
`SMC`	Use squared multiple correlations (SMC=TRUE) or use 1 as initial communality estimate. Try using 1 if imaginary eigen values are reported.
`missing`	if scores are TRUE, and missing=TRUE, then impute missing values using either the median or the mean
`impute`	"median" or "mean" values are used to replace missing values
`min.err`	Iterate until the change in communalities is less than min.err
`digits`	How many digits of output should be returned
`max.iter`	Maximum number of iterations for convergence
`symmetric`	symmetric=TRUE forces symmetry by just looking at the lower off diagonal values
`warnings`	warnings=TRUE => warn if number of factors is too many

Details

Factor analysis is an attempt to approximate a correlation or covariance matrix with one of lesser rank. The basic model is that ${}_nR_n \approx_n F_{kk}F'_n + U^2$ where k is much less than n. There are many ways to do factor analysis, and maximum likelihood procedures are probably the most preferred (see `factanal`). The existence of uniquenesses is what distinguishes factor analysis from principal components analysis (e.g., `principal`).

Principal axes factor analysis has a long history in exploratory analysis and is a straightforward procedure. Successive eigen value decompositions are done on a correlation matrix with the diagonal replaced with diag (FF') until sum(diag(FF')) does not change (very much). The current limit of max.iter =50 seems to work for most problems, but the Holzinger-Harmon 24 variable problem needs about 203 iterations to converge for a 5 factor solution.

Principal axes may be used in cases when maximum likelihood solutions fail to converge.

A problem in factor analysis is to find the best estimate of the original communalities. Using the Squared Multiple Correlation (SMC) for each variable will underestimate the communalities, using 1s will over estimate. By default, the SMC estimate is used. If, however, a solution fails to be achieved, it is useful to try again using ones (SMC =FALSE).

The algorithm does not attempt to find the best (as defined by a maximum likelihood criterion) solution, but rather one that converges rapidly using successive eigen value decompositions. The maximum likelihood criterion of fit and the associated chi square value are reported, and will be worse than that found using maximum likelihood procedures.

Although for items, it is typical to find factor scores by scoring the salient items (using, e.g.,`score.items` factor scores can be estimated by regression.

Value

`values`	Eigen values of the final solution
`communality`	Communality estimates for each item. These are merely the sum of squared factor loadings for that item.
`rotation`	which rotation was requested?
`n.obs`	number of observations specified or found
`loadings`	An item by factor loading matrix of class "loadings" Suitable for use in other programs (e.g., GPA rotation or factor2cluster.
`fit`	How well does the factor model reproduce the correlation matrix. (See `VSS`, `ICLUST`, and `principal` for this fit statistic.
`fit.off`	how well are the off diagonal elements reproduced?
`dof`	Degrees of Freedom for this model. This is the number of observed correlations minus the number of independent parameters. Let n=Number of items, nf = number of factors then $dof = n*(n-1)/2 - n*nf + nf*(nf-1)/2$
`objective`	value of the function that is minimized by maximum likelihood procedures. This is reported for comparison purposes and as a way to estimate chi square goodness of fit. The objective function is $f = log(trace((FF' + U2)^{-1}R) - log(\|(FF' + U2)^{-1}R\|) - n.items.$

`STATISTIC` If the number of observations is specified or found, this is a chi square based upon the objective function, f. Using the formula from `factanal`(which seems to be Bartlett's test) :
$\chi^2 = (n.obs - 1 - (2 * p + 5)/6 - (2 * factors)/3)) * f$

`PVAL` If n.obs > 0, then what is the probability of observing a chisquare this large or larger?

`Phi` If oblique rotations (using oblimin from the GPArotation package or promax) are requested, what is the interfactor correlation.

`communality.iterations`
The history of the communality estimates. Probably only useful for teaching what happens in the process of iterative fitting.

`residual` If residuals are requested, this is the matrix of residual correlations after the factor model is applied.

Author(s)

William Revelle

References

Gorsuch, Richard, (1983) Factor Analysis. Lawrence Erlebaum Associates.

Revelle, William. (in prep) An introduction to psychometric theory with applications in R. Springer. Working draft available at `http://personality-project.org/r/book.html`

See Also

`principal`, `VSS`, `ICLUST`

Examples

```
#using the Harman 24 mental tests, compare a principal factor with a principal components solution
pc <- principal(Harman74.cor$cov,4,rotate="varimax")
pa <- factor.pa(Harman74.cor$cov,4,rotate="varimax")
round(factor.congruence(pc,pa),2)

#then compare with a maximum likelihood solution using factanal
mle <- factanal(x,4,covmat=Harman74.cor$cov)
round(factor.congruence(mle,pa),2)
#note that the order of factors and the sign of some of factors differ

#finally, compare the unrotated factor and pca solutions
pc1 <- principal(Harman74.cor$cov,4,rotate="none")
pa1 <- factor.pa(Harman74.cor$cov,4,rotate="none")
round(factor.congruence(pc1,pa1),2)
#note that the order of factors and the sign of some of factors differ
```

`factor.residuals` $R^* = R - F\ F'$

Description

The basic factor or principal components model is that a correlation or covariance matrix may be reproduced by the product of a factor loading matrix times its transpose. Find the residuals of the original minus the reproduced matrix. Used by `factor.fit`, `VSS`, `ICLUST`, etc.

Usage

```
factor.residuals(r, f)
```

Arguments

`r`	A correlation matrix
`f`	A factor model matrix or a list of class loadings

Details

The basic factor equation is ${}_nR_n \approx_n F_{kk}F'_n + U^2$. Residuals are just R* = R - F'F. The residuals should be (but in practice probably rarely are) examined to understand the adequacy of the factor analysis. When doing Factor analysis or Principal Components analysis, one usually continues to extract factors/components until the residuals do not differ from those expected from a random matrix.

Value

rstar is the residual correlation matrix.

Author(s)

Maintainer: William Revelle <revelle@northwestern.edu>

See Also

`factor.pa`, `principal`, `VSS`, `ICLUST`

Examples

```
fa2 <- factor.pa(Harman74.cor$cov,2,rotate=TRUE)
 fa2resid <- factor.residuals(Harman74.cor$cov,fa2)
 fa2resid[1:4,1:4] #residuals with two factors extracted
 fa4 <- factor.pa(Harman74.cor$cov,4,rotate=TRUE)
 fa4resid <- factor.residuals(Harman74.cor$cov,fa4)
 fa4resid[1:4,1:4] #residuals with 4 factors extracted
```

`factor.rotate` *"Hand" rotate a factor loading matrix*

Description

Given a factor or components matrix, it is sometimes useful to do arbitrary rotations of particular pairs of variables. This supplements the much more powerful rotation package GPArotation and is meant for specific requirements to do unusual rotations.

Usage

```
factor.rotate(f, angle, col1=1, col2=2)
```

Arguments

`f`	original loading matrix or a data frame (can be output from a factor analysis function
`angle`	angle (in degrees!) to rotate
`col1`	column in factor matrix defining the first variable
`col2`	column in factor matrix defining the second variable

Details

Partly meant as a demonstration of how rotation works, factor.rotate is useful for those cases that require specific rotations that are not available in more advanced packages such as GPArotation.

Value

the resulting rotated matrix of loadings.

Note

For a complete rotation package, see GPArotation

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu ⟩

References

`http://personality-project.org/revelle/syllabi/405.syllabus.html`

Examples

```
#using the Harman 24 mental tests, rotate the 2nd and 3rd factors 45 degrees
pc <- principal(Harman74.cor$cov,4,rotate=TRUE)
pcr45 <- factor.rotate(pc,45,2,3)
pcr90 <- factor.rotate(pcr45,45,2,3)
print(factor.congruence(pc,pcr45),digits=3) #poor congruence with original
print(factor.congruence(pc,pcr90),digits=3) #factor 2 and 3 have been exchanged and 3 flipped
```

---

factor2cluster *Extract cluster definitions from factor loadings*

Description

Given a factor or principal components loading matrix, assign each item to a cluster corresponding to the largest (signed) factor loading for that item. Essentially, this is a Very Simple Structure approach to cluster definition that corresponds to what most people actually do: highlight the largest loading for each item and ignore the rest.

Usage

```
factor2cluster(loads, cut = 0)
```

Arguments

`loads`	either a matrix of loadings, or the result of a factor analysis/principal components analyis with a loading component
`cut`	Extract items with absolute loadings > cut

Details

A factor/principal components analysis loading matrix is converted to a cluster (-1,0,1) definition matrix where each item is assigned to one and only one cluster. This is a fast way to extract items that will be unit weighted to form cluster composites. Use this function in combination with cluster.cor to find the corrleations of these composite scores.

A typical use in the SAPA project is to form item composites by clustering or factoring (see `ICLUST`, `principal`), extract the clusters from these results (`factor2cluster`), and then form the composite correlation matrix using `cluster.cor`. The variables in this reduced matrix may then be used in multiple R procedures using mat.regress.

The input may be a matrix of item loadings, or the output from a factor analysis which includes a loadings matrix.

Value

a matrix of -1,0,1 cluster definitions for each item.

Author(s)

http://personality-project.org/revelle.html

Maintainer: William Revelle ⟨ revelle@northwestern.edu ⟩

References

http://personality-project.org/r/r.vss.html

See Also

cluster.cor, factor2cluster, factor.pa, principal, ICLUST

Examples

```
## Not run:
f  <- factanal(x,4,covmat=Harman74.cor$cov)
factor2cluster(f)
## End(Not run)
#                        Factor1 Factor2 Factor3 Factor4
#VisualPerception              0       1       0       0
#Cubes                         0       1       0       0
#PaperFormBoard                0       1       0       0
#Flags                         0       1       0       0
#GeneralInformation            1       0       0       0
#PargraphComprehension         1       0       0       0
#SentenceCompletion            1       0       0       0
#WordClassification            1       0       0       0
#WordMeaning                   1       0       0       0
#Addition                      0       0       1       0
#Code                          0       0       1       0
#CountingDots                  0       0       1       0
#StraightCurvedCapitals        0       0       1       0
#WordRecognition               0       0       0       1
#NumberRecognition             0       0       0       1
#FigureRecognition             0       0       0       1
#ObjectNumber                  0       0       0       1
#NumberFigure                  0       0       0       1
#FigureWord                    0       0       0       1
#Deduction                     0       1       0       0
#NumericalPuzzles              0       0       1       0
#ProblemReasoning              0       1       0       0
#SeriesCompletion              0       1       0       0
#ArithmeticProblems            0       0       1       0
```

`fisherz` *Fisher r to z and z to r and confidence intervals*

Description

convert a correlation to a z score or z to r using the Fisher transformation or find the confidence intervals for a specified correlation

Usage

```
fisherz(rho)
fisherz2r(z)
r.con(rho,n,p=.95,twotailed=TRUE)
r2t(rho,n)
```

Arguments

`rho`	a Pearson r
`z`	A Fisher z
`n`	Sample size for confidence intervals
`p`	Confidence interval
`twotailed`	Treat p as twotailed p

Value

z value corresponding to r (fisherz) \ r corresponding to z (fisherz2r) \ lower and upper p confidence intervals (r.con) \ t with n-2 df (r2t)

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu ⟩

Examples

```
cors <- seq(-.9,.9,.1)
zs <- fisherz(cors)
rs <- fisherz2r(zs)
round(zs,2)
 n <- 30
 r <- seq(0,.9,.1)
 rc <- matrix(r.con(r,n),ncol=2)
t <- r*sqrt(n-2)/sqrt(1-r^2)
 p <- (1-pt(t,n-2))/2
 r.rc <- r.rc <- data.frame(r=r,z=fisherz(r),lower=rc[,1],upper=rc[,2],t=t,p=p)
 round(r.rc,2)
```

`galton` *Galton's Mid parent child height data*

Description

Two of the earliest examples of the correlation coefficient were Francis Galton's data sets on the relationship between mid parent and child height and the similarity of parent generation peas with child peas. This is the data set for the Galton height.

Usage

```
data(galton)
```

Format

A data frame with 928 observations on the following 2 variables.

`parent` Mid Parent heights (in inches)

`child` Child Height

Details

Female heights were adjusted by 1.08 to compensate for sex differences. (This was done in the original data set)

Source

This is just the galton data set from UsingR, slightly rearranged.

References

Stigler, S. M. (1999). Statistics on the Table: The History of Statistical Concepts and Methods. Harvard University Press. Galton, F. (1869). Hereditary Genius: An Inquiry into its Laws and Con- sequences. London: Macmillan.

Wachsmuth, A.W., Wilkinson L., Dallal G.E. (2003). Galton's bend: A previously undiscovered nonlinearity in Galton's family stature regression data. The American Statistician, 57, 190-1922.

Examples

```
data(galton)
describe(galton)
pairs.panels(galton,lm=TRUE)
```

`geometric.mean` *Find the geometric mean of a vector or columns of a data.frame.*

Description

The geometric mean is the nth root of n products or e to the mean log of x. Useful for describing non-normal, i.e., geometric distributions.

Usage

```
geometric.mean(x)
```

Arguments

`x`	a vector or data.frame

Details

Useful for teaching how to write functions, also useful for showing the different ways of estimating central tendency.

Value

geometric mean(s) of x or x.df.

Note

Not particularly useful if there are elements that are <= 0.

Author(s)

William Revelle

See Also

`harmonic.mean`, `mean`

Examples

```
x <- seq(1,5)
x2 <- x^2
geometric.mean(x)
geometric.mean(x2)
```

`guttman` *Alternative estimates of test reliabiity*

Description

Eight alternative estimates of test reliability include the six discussed by Guttman (1945), four discussed by ten Berge and Zergers (1978) ($\mu_0 \ldots \mu_3$) as well as β (the worst split half, Revelle, 1979), the glb (greatest lowest bound) discussed by Bentler and Woodward (1980), and ω_h and ω_t (McDonald, 1999; Zinbarg et al., 2005).

Usage

```
guttman(r,key=NULL,digits=2)
tenberge(r,digits=2)
glb(r,key=NULL,digits=2)
```

Arguments

`r`	A correlation matrix or raw data matrix.
`key`	a vector of -1, 0, 1 to select or reverse items
`digits`	How many digits of accuracy in the output?

Details

Surprisingly, 104 years after Spearman (1904) introduced the concept of reliability to psychologists, there are still multiple approaches for measuring it. Although very popular, Cronbach's α (1951) underestimates the reliability of a test and over estimates the first factor saturation. The guttman function includes the six estimates discussed by Guttman (1945), four of ten Berge and Zergers (1978), as well as Revelle's β (1979) using `ICLUST`. The companion function, `omega` calculates omega hierarchical (ω_h) and omega total (ω_t).

$$\lambda_1 = 1 - \frac{tr(\vec{V}_x)}{V_x} = \frac{V_x - tr(\vec{V}_x)}{V_x}$$

The second bound, λ_2 replaces the diagonal with a function of the square root of the sums of squares of the off diagonal elements. Let $C_2 = \vec{1}(\vec{V} - diag(\vec{V}))^2\vec{1}'$, then

$$\lambda_2 = \lambda_1 + \frac{\sqrt{\frac{n}{n-1}C_2}}{V_x} = \frac{V_x - tr(\vec{V}_x) + \sqrt{\frac{n}{n-1}C_2}}{V_x}.$$

Effectively, this is replacing the diagonal with n * the square root of the average squared off diagonal element.

Guttman's 3rd lower bound, λ_3, also modifies λ_1 and estimates the true variance of each item as the average covariance between items and is, of course, the same as Cronbach's α.

$$\lambda_3 = \lambda_1 + \frac{\frac{V_X - tr(\vec{V}_X)}{n(n-1)}}{V_X} = \frac{n\lambda_1}{n-1} = \frac{n}{n-1}\Big(1 - \frac{tr(\vec{V})_x}{V_x}\Big) = \frac{n}{n-1}\frac{V_x - tr(\vec{V}_x)}{V_x} = \alpha$$

This is just replacing the diagonal elements with the average off diagonal elements. $\lambda_2 \geq \lambda_3$ with $\lambda_2 > \lambda_3$ if the covariances are not identical.

λ_3 and λ_2 are both corrections to λ_1 and this correction may be generalized as an infinite set of successive improvements. (Ten Berge and Zegers, 1978)

$$\mu_r = \frac{1}{V_x}\big(p_o + (p_1 + (p_2 + \ldots (p_{r-1} + (p_r)^{1/2})^{1/2} \ldots)^{1/2})^{1/2}\big), r = 0, 1, 2, \ldots$$

where

$$p_h = \sum_{i \neq j} \sigma_{ij}^{2h}, h = 0, 1, 2, \ldots r-1$$

and

$$p_h = \frac{n}{n-1}\sigma_{ij}^{2h}, h = r$$

tenberge & Zegers (1978). Clearly $\mu_0 = \lambda_3 = \alpha$ and $\mu_1 = \lambda_2$. $\mu_r \geq \mu_{r-1} \geq \ldots \mu_1 \geq \mu_0$, although the series does not improve much after the first two steps.

Guttman's fourth lower bound, λ_4 was originally proposed as any spit half reliability but has been interpreted as the greatest split half reliability. If $\vec{X}$ is split into two parts, $\vec{X}_a$ and $\vec{X}_b$, with correlation r_{ab} then

$$\lambda_4 = 2\Big(1 - \frac{V_{X_a} + V_{X_b}}{V_X}\Big) = \frac{4r_{ab}}{V_x} = \frac{4r_{ab}}{V_{X_a} + V_{X_b} + 2r_{ab}V_{X_a}V_{X_b}}$$

which is just the normal split half reliability, but in this case, of the most similar splits.

λ_5, Guttman's fifth lower bound, replaces the diagonal values with twice the square root of the maximum (across items) of the sums of squared interitem covariances

$$\lambda_5 = \lambda_1 + \frac{2\sqrt{\bar{C}_2}}{V_X}.$$

Although superior to λ_1, λ_5 underestimates the correction to the diagonal. A better estimate would be analogous to the correction used in λ_3:

$$\lambda_{5+} = \lambda_1 + \frac{n}{n-1}\frac{2\sqrt{\bar{C}_2}}{V_X}.$$

Guttman's final bound considers the amount of variance in each item that can be accounted for the linear regression of all of the other items (the squared multiple correlation or smc), or more precisely, the variance of the errors, e_j^2, and is

$$\lambda_6 = 1 - \frac{\sum e_j^2}{V_x} = 1 - \frac{\sum(1 - r_{smc}^2)}{V_x}$$

Guttman's λ_4 is the greatest split half reliability. This is found by combining the output from three different approaches, and seems to work for all test cases yet tried. Lambda 4 is reported as the max of these three algorithms.

The algorithms are

a) Do an ICLUST of the reversed correlation matrix. ICLUST normally forms the most distinct clusters. By reversing the correlations, it will tend to find the most related cluster. Truly a weird approach but tends to work.

b) Alternatively, a kmeans clustering of the correlations (with the diagonal replaced with 0 to make pseudo distances) can produce 2 similar clusters.

c) Clusters identified by assigning items to two clusters based upon their order on the first principal factor. (Highest to cluster 1, next 2 to cluster 2, etc.)

Value

`beta`	The normal beta estimate of cluster similarity from ICLUST. This is an estimate of the general factor saturation.
`tenberge$mu1`	tenBerge mu 1 is functionally alpha
`tenberge$mu2`	one of the sequence of estimates mu1 ... mu3
`beta.factor`	For experimental purposes, what is the split half based upon the two factor solution?
`glb.IC`	Greatest split half based upon ICLUST of reversed correlations
`glb.Km`	Greatest split half based upon a kmeans clustering.
`glb.Fa`	Greatest split half based upon the items assigned by factor analysis.
`glb.max`	max of the above estimates
`keys`	scoring keys from each of the alternative methods of forming best splits

Author(s)

William Revelle

References

Cronbach, L.J. (1951) Coefficient alpha and the internal strucuture of tests. Psychometrika, 16, 297-334.

Guttman, L. (1945). A basis for analyzing test-retest reliability. Psychometrika, 10 (4), 255-282.

Revelle, W. (1979). Hierarchical cluster-analysis and the internal structure of tests. Multivariate Behavioral Research, 14 (1), 57-74.

Revelle, W. and Zinbarg, R. E. (2009) Coefficients alpha, beta, omega and the glb: comments on Sijtsma. Psychometrika, 2009.

Ten Berge, J. M. F., & Zegers, F. E. (1978). A series of lower bounds to the reliability of a test. Psychometrika, 43 (4), 575-579.

Zinbarg, R. E., Revelle, W., Yovel, I., & Li, W. (2005). Cronbach's α , Revelle's β , and McDonald's ω_h): Their relations with each other and two alternative conceptualizations of reliability. Psychometrika, 70 (1), 123-133.

See Also

`omega`, `ICLUST`,

Examples

```
data(attitude)
glb(attitude)
guttman(attitude)
```

`harmonic.mean` *Find the harmonic mean of a vector, matrix, or columns of a data.frame*

Description

The harmonic mean is merely the reciprocal of the arithmetic mean of the reciprocals.

Usage

```
harmonic.mean(x)
```

Arguments

`x`	a vector, matrix, or data.frame

Details

Included as an example for teaching about functions. As well as for a discussion of how to estimate central tendencies.

Value

The harmonic mean(s)

Note

Included as a simple demonstration of how to write a function

Examples

```
x <- seq(1,5)
x2 <- x^2
harmonic.mean(x)
harmonic.mean(x2)
```

`headtail` *Combine calls to head and tail*

Description

A quick way to show the first and last n lines of a data.frame, matrix, or a textt object. Just a pretty call to `head` and `tail`

Usage

```
headtail(x,hlength=4,tlength=4,digits=2)
```

Arguments

`x`	A matrix or data frame or free text
`hlength`	The number of lines at the beginning to show
`tlength`	The number of lines at the end to show
`digits`	Round off the data to digits

Value

The first hlength and last tlength lines of a matrix or data frame with an ellipsis in between. If the input is neither a matrix nor data frame, the output will be the first hlength and last tlength lines.

See Also

`head` and `tail`

Examples

```
x <- matrix(sample(10,1000,TRUE),ncol=5)
headtail(x,4,8)
```

`heights` *A data.frame of the Galton (1888) height and cubit data set.*

Description

Francis Galton introduced the 'co-relation' in 1888 with a paper discussing how to measure the relationship between two variables. His primary example was the relationship between height and forearm length. The data table (`cubits`) is taken from Galton (1888). Unfortunately, there seem to be some errors in the original data table in that the marginal totals do not match the table.

The data frame, `heights`, is converted from this table using `table2df`.

Usage

```
data(heights)
```

Format

A data frame with 348 observations on the following 2 variables.

`height` Height in inches

`cubit` Forearm length in inches

Details

Sir Francis Galton (1888) published the first demonstration of the correlation coefficient. The regression (or reversion to mediocrity) of the height to the length of the left forearm (a cubit) was found to .8. The original table `cubits` is taken from Galton (1888). There seem to be some errors in the table as published in that the row sums do not agree with the actual row sums. These data are used to create a matrix using `table2matrix` for demonstrations of analysis and displays of the data.

Source

Galton (1888)

References

Galton, Francis (1888) Co-relations and their measurement. Proceedings of the Royal Society. London Series,45,135-145,

See Also

`table2matrix`, `table2df`, `cubits`, `ellipses`, `galton`

Examples

```
data(heights)
ellipses(heights,n=1,main="Galton's co-relation data set")
```

ICC — *Intraclass Correlations (ICC1, ICC2, ICC3 from Shrout and Fleiss)*

Description

The Intraclass correlation is used as a measure of association when studying the reliability of raters. Shrout and Fleiss (1979) outline 6 different estimates, that depend upon the particular experimental design. All are implemented and given confidence limits.)

Usage

```
ICC(x,digits=2,alpha=.05)
```

Arguments

`x`	a matrix or dataframe of ratings
`digits`	Round the output to digits
`alpha`	The alpha level for significance

Details

Shrout and Fleiss (1979) consider six cases of reliability of ratings done by k raters on n targets.

ICC1) Each target is rated by a different judge and the judges are selected at random. (This is a one-way ANOVA random effects model.) 2) A random sample of k judges rate each target. The measure is one of absolute agreement in the ratings. 3) A fixed set of k judges rate each target. There is no generalization to a larger population of judges.

Then, for each of these cases, is reliability to be estimated for a single rating or for the average of k ratings? (The 1 rating case is equivalent to the average intercorrelation, the k rating case to the Spearman Brown adjusted reliability.)

ICC1 is sensitive to differences in means between raters and is a measure of absolute agreement.

ICC2 and ICC3 remove mean differences between judges, but are sensitive to interactions of raters by judges. The difference between ICC2 and ICC3 is whether raters are seen as fixed or random effects.

ICC1k, ICC2k, ICC3K reflect the means of k raters.

Value

`results`	A matrix of 6 rows and 8 columns, including the ICCs, F test, p values, and confidence limits

Note

The results for the Lower and Upper Bounds for ICC(2,k) do not match those of SPSS 9 or 10, but do match the definitions of Shrout and Fleiss. SPSS seems to be using the formula in McGraw and Wong, but not the errata on p 390. They seem to have fixed it in more recent releases.

Author(s)

William Revelle

References

Shrout, Patrick E. and Fleiss, Joseph L. Intraclass correlations: uses in assessing rater reliability. Psychological Bulletin, 1979, 86, 420-3428.

McGraw, Kenneth O. and Wong, S. P. (1996), Forming inferences about some intraclass correlation coefficients. Psychological Methods, 1, 30-46. + errata on page 390.

Revelle, W. (in prep) An introduction to psychometric theory with applications in R. Springer. (working draft available at `http://personality-project.org/r/book/`

Examples

```
sf <- matrix(c(9,    2,   5,    8,
6,    1,   3,    2,
8,    4,   6,    8,
7,    1,   2,    6,
10,   5,   6,    9,
6,   2,   4,    7),ncol=4,byrow=TRUE)
colnames(sf) <- paste("J",1:4,sep="")
rownames(sf) <- paste("S",1:6,sep="")
sf  #example from Shrout and Fleiss (1979)
ICC(sf)
```

ICLUST.cluster *Function to form hierarchical cluster analysis of items*

Description

The guts of the `ICLUST` algorithm. Called by `ICLUST` See ICLUST for description.

Usage

```
ICLUST.cluster(r.mat, ICLUST.options)
```

Arguments

`r.mat`	A correlation matrix
`ICLUST.options`	A list of options (see `ICLUST`)

Details

See `ICLUST`

Value

A list of cluster statistics, described more fully in `ICLUST`

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

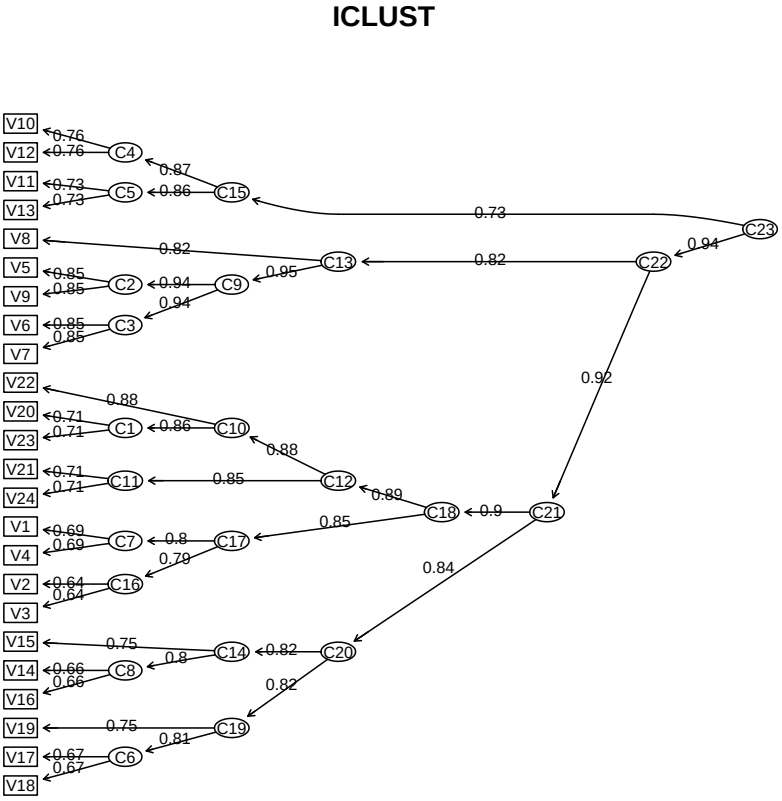

Figure 5: The Harman 24 variables may also be cluster analyzed to highlight the hierarchical structure of abilities. Compare to the hierarchical factor solution using omega (Figure 6) or the orthogonal factor solution (Figure 4)

Note

Although the main code for ICLUST is here in ICLUST.cluster, the more extensive documentation is for ICLUST.

Author(s)

William Revelle

References

Revelle, W. 1979, Hierarchical Cluster Analysis and the Internal Structure of Tests. Multivariate Behavioral Research, 14, 57-74. http://personality-project.org/revelle/publications/iclust.pdf
See also more extensive documentation at http://personality-project.org/r/r.ICLUST.html

See Also

ICLUST.graph,ICLUST, cluster.fit , VSS, omega

ICLUST.graph *create control code for ICLUST graphical output*

Description

Given a cluster structure determined by ICLUST, create dot code to describe the ICLUST output. To use the dot code, use either http://www.graphviz.org/ Graphviz or a commercial viewer (e.g., OmniGraffle).

Usage

```
ICLUST.graph(ic.results, out.file,min.size=1, short = FALSE,labels=NULL,
size = c(8, 6), node.font = c("Helvetica", 14), edge.font = c("Helvetica", 12),
 rank.direction=c("RL","TB","LR","BT"), digits = 2, title = "ICLUST", ...)
```

Arguments

ic.results	output list from ICLUST
out.file	name of output file (defaults to console)
min.size	draw a smaller node (without all the information) for clusters < min.size – useful for large problems
short	if short==TRUE, don't use variable names
labels	vector of text labels (contents) for the variables
size	size of output
node.font	Font to use for nodes in the graph
edge.font	Font to use for the labels of the arrows (edges)

rank.direction	LR or RL
digits	number of digits to show
title	any title
...	other options to pass

Details

Will create (or overwrite) an output file and print out the dot code to show a cluster structure. This dot file may be imported directly into a dot viewer (e.g., http://www.graphviz.org/). The "dot" language is a powerful graphic description language that is particulary appropriate for viewing cluster output. Commercial graphics programs (e.g., OmniGraffle) can also read (and clean up) dot files.

ICLUST.graph takes the output from ICLUST results and processes it to provide a pretty picture of the results. Original variables shown as rectangles and ordered on the left hand side (if rank direction is RL) of the graph. Clusters are drawn as ellipses and include the alpha, beta, and size of the cluster. Edges show the cluster intercorrelations.

It is possible to trim the output to not show all cluster information. Clusters < min.size are shown as small ovals without alpha, beta, and size information.

Value

Output is a set of dot commands written either to console or to the output file. These commands may then be used as input to any "dot" viewer, e.g., Graphviz.

Author(s)

⟨revelle@northwestern.edu ⟩
http://personality-project.org/revelle.html

References

ICLUST: http://personality-project.org/r/r.iclust.html

See Also

VSS.plot, ICLUST

Examples

```
## Not run:
test.data <- Harman74.cor$cov
ic.out <- ICLUST(test.data)
out.file <- file.choose(new=TRUE)   #create a new file to write the plot commands to
ICLUST.graph(ic.out,out.file)
now go to graphviz (outside of R) and open the out.file you created
print(ic.out,digits=2)
## End(Not run)
```

```
#test.data <- Harman74.cor$cov
#my.iclust <- ICLUST(test.data)
#ICLUST.graph(my.iclust)
#
#
#digraph ICLUST {
#  rankdir=RL;
#  size="8,8";
#  node [fontname="Helvetica" fontsize=14 shape=box, width=2];
#  edge [fontname="Helvetica" fontsize=12];
# label = "ICLUST";
#       fontsize=20;
#V1  [label = VisualPerception];
#V2  [label = Cubes];
#V3  [label = PaperFormBoard];
#V4  [label = Flags];
#V5  [label = GeneralInformation];
#V6  [label = PargraphComprehension];
#V7  [label = SentenceCompletion];
#V8  [label = WordClassification];
#V9  [label = WordMeaning];
#V10  [label = Addition];
#V11  [label = Code];
#V12  [label = CountingDots];
#V13  [label = StraightCurvedCapitals];
#V14  [label = WordRecognition];
#V15  [label = NumberRecognition];
#V16  [label = FigureRecognition];
#V17  [label = ObjectNumber];
#V18  [label = NumberFigure];
#V19  [label = FigureWord];
#V20  [label = Deduction];
#V21  [label = NumericalPuzzles];
#V22  [label = ProblemReasoning];
#V23  [label = SeriesCompletion];
#V24  [label = ArithmeticProblems];
#node [shape=ellipse, width ="1"];
#C1-> V9 [ label = 0.78 ];
#C1-> V5 [ label = 0.78 ];
#C2-> V12 [ label = 0.66 ];
#C2-> V10 [ label = 0.66 ];
#C3-> V18 [ label = 0.53 ];
#C3-> V17 [ label = 0.53 ];
#C4-> V23 [ label = 0.59 ];
#C4-> V20 [ label = 0.59 ];
#C5-> V13 [ label = 0.61 ];
#C5-> V11 [ label = 0.61 ];
#C6-> V7 [ label = 0.78 ];
#C6-> V6 [ label = 0.78 ];
#C7-> V4 [ label = 0.55 ];
#C7-> V1 [ label = 0.55 ];
#C8-> V16 [ label = 0.5 ];
#C8-> V14 [ label = 0.49 ];
```

```
#C9-> C1 [ label = 0.86 ];
#C9-> C6 [ label = 0.86 ];
#C10-> C4 [ label = 0.71 ];
#C10-> V22 [ label = 0.62 ];
#C11-> V21 [ label = 0.56 ];
#C11-> V24 [ label = 0.58 ];
#C12-> C10 [ label = 0.76 ];
#C12-> C11 [ label = 0.67 ];
#C13-> C8 [ label = 0.61 ];
#C13-> V15 [ label = 0.49 ];
#C14-> C2 [ label = 0.74 ];
#C14-> C5 [ label = 0.72 ];
#C15-> V3 [ label = 0.48 ];
#C15-> C7 [ label = 0.65 ];
#C16-> V19 [ label = 0.48 ];
#C16-> C3 [ label = 0.64 ];
#C17-> V8 [ label = 0.62 ];
#C17-> C12 [ label = 0.8 ];
#C18-> C17 [ label = 0.82 ];
#C18-> C15 [ label = 0.68 ];
#C19-> C16 [ label = 0.66 ];
#C19-> C13 [ label = 0.65 ];
#C20-> C19 [ label = 0.72 ];
#C20-> C18 [ label = 0.83 ];
#C21-> C20 [ label = 0.87 ];
#C21-> C9 [ label = 0.76 ];
#C22-> 0 [ label = 0 ];
#C22-> 0 [ label = 0 ];
#C23-> 0 [ label = 0 ];
#C23-> 0 [ label = 0 ];
#C1  [label =   "C1\n  alpha= 0.84\n beta=  0.84\nN= 2"] ;
#C2  [label =   "C2\n  alpha= 0.74\n beta=  0.74\nN= 2"] ;
#C3  [label =   "C3\n  alpha= 0.62\n beta=  0.62\nN= 2"] ;
#C4  [label =   "C4\n  alpha= 0.67\n beta=  0.67\nN= 2"] ;
#C5  [label =   "C5\n  alpha= 0.7\n beta=  0.7\nN= 2"] ;
#C6  [label =   "C6\n  alpha= 0.84\n beta=  0.84\nN= 2"] ;
#C7  [label =   "C7\n  alpha= 0.64\n beta=  0.64\nN= 2"] ;
#C8  [label =   "C8\n  alpha= 0.58\n beta=  0.58\nN= 2"] ;
#C9  [label =   "C9\n  alpha= 0.9\n beta=  0.87\nN= 4"] ;
#C10  [label =   "C10\n  alpha= 0.74\n beta=  0.71\nN= 3"] ;
#C11  [label =   "C11\n  alpha= 0.62\n beta=  0.62\nN= 2"] ;
#C12  [label =   "C12\n  alpha= 0.79\n beta=  0.74\nN= 5"] ;
#C13  [label =   "C13\n  alpha= 0.64\n beta=  0.59\nN= 3"] ;
#C14  [label =   "C14\n  alpha= 0.79\n beta=  0.74\nN= 4"] ;
#C15  [label =   "C15\n  alpha= 0.66\n beta=  0.58\nN= 3"] ;
#C16  [label =   "C16\n  alpha= 0.65\n beta=  0.57\nN= 3"] ;
#C17  [label =   "C17\n  alpha= 0.81\n beta=  0.71\nN= 6"] ;
#C18  [label =   "C18\n  alpha= 0.84\n beta=  0.75\nN= 9"] ;
#C19  [label =   "C19\n  alpha= 0.74\n beta=  0.65\nN= 6"] ;
#C20  [label =   "C20\n  alpha= 0.87\n beta=  0.74\nN= 15"] ;
#C21  [label =   "C21\n  alpha= 0.9\n beta=  0.77\nN= 19"] ;
#C22  [label =   "C22\n  alpha= 0\n beta=  0\nN= 0"] ;
#C23  [label =   "C23\n  alpha= 0\n beta=  0\nN= 0"] ;
```

```
#{ rank=same;
#V1;V2;V3;V4;V5;V6;V7;V8;V9;V10;V11;V12;V13;V14;V15;V16;V17;V18;V19;V20;V21;V22;V23;V24;}}
#
#copy the above output to Graphviz and draw it
#see \url{http://personality-project.org/r/r.ICLUST.html} for an example.
```

`ICLUST.rgraph` *Draw an ICLUST graph using the Rgraphviz package*

Description

Given a cluster structure determined by `ICLUST`, create a rgraphic directly using Rgraphviz. To create dot code to describe the `ICLUST` output with more precision, use `ICLUST.graph`. As an option, dot code is also generated and saved in a file. To use the dot code, use either `http://www.graphviz.org/` Graphviz or a commercial viewer (e.g., OmniGraffle).

Usage

```
ICLUST.rgraph(ic.results, out.file = NULL, min.size = 1, short = FALSE, labels = NULL, size = c
```

Arguments

`ic.results`	output list from ICLUST
`out.file`	File name to save optional dot code.
`min.size`	draw a smaller node (without all the information) for clusters < min.size – useful for large problems
`short`	if short==TRUE, don't use variable names
`labels`	vector of text labels (contents) for the variables
`size`	size of output
`node.font`	Font to use for nodes in the graph
`edge.font`	Font to use for the labels of the arrows (edges)
`rank.direction`	LR or TB or RL
`digits`	number of digits to show
`title`	any title
`label.font`	The variable labels can be a different size than the other nodes. This is particularly helpful if the number of variables is large or the labels are long.
`...`	other options to pass

Details

Will create (or overwrite) an output file and print out the dot code to show a cluster structure. This dot file may be imported directly into a dot viewer (e.g., `http://www.graphviz.org/`). The "dot" language is a powerful graphic description language that is particulary appropriate for viewing cluster output. Commercial graphics programs (e.g., OmniGraffle) can also read (and clean up) dot files.

ICLUST.rgraph takes the output from `ICLUST` results and processes it to provide a pretty picture of the results. Original variables shown as rectangles and ordered on the left hand side (if rank direction is RL) of the graph. Clusters are drawn as ellipses and include the alpha, beta, and size of the cluster. Edges show the cluster intercorrelations.

It is possible to trim the output to not show all cluster information. Clusters < min.size are shown as small ovals without alpha, beta, and size information.

Value

Output is a set of dot commands written either to console or to the output file. These commands may then be used as input to any "dot" viewer, e.g., Graphviz.

ICLUST.rgraph is a version of `ICLUST.graph` that uses Rgraphviz to draw on the screen as well.

Additional output is drawn to main graphics screen.

Note

Requires Rgraphviz

Author(s)

⟨revelle@northwestern.edu ⟩
`http://personality-project.org/revelle.html`

References

ICLUST: `http://personality-project.org/r/r.iclust.html`

See Also

`VSS.plot`, `ICLUST`

Examples

```
test.data <- Harman74.cor$cov

if(require(Rgraphviz) ) {ic.out <- ICLUST(test.data,labels=colnames(test.data)) }
```

ICLUST.sort *Sort items by absolute size of cluster loadings*

Description

Given a cluster analysis or factor analysis loadings matrix, sort the items by the (absolute) size of each column of loadings. Used as part of ICLUST and SAPA analyses.

Usage

```
ICLUST.sort(ic.load, cut = 0, labels = NULL,keys=FALSE)
```

Arguments

ic.load	The output from a factor or principal components analysis, or from ICLUST, or a matrix of loadings.
cut	Do not include items in clusters with absolute loadings less than cut
labels	labels for each item.
keys	should cluster keys be returned? Useful if clusters scales are to be scored.

Details

When interpreting cluster or factor analysis outputs, is is useful to group the items in terms of which items have their biggest loading on each factor/cluster and then to sort the items by size of the absolute factor loading.

A stable cluster solution will be one in which the output of these cluster definitions does not vary when clusters are formed from the clusters so defined.

With the keys=TRUE option, the resulting cluster keys may be used to score the original data or the correlation matrix to form clusters from the factors.

Value

sorted	A data.frame of item numbers, item contents, and item x factor loadings.
cluster	A matrix of -1, 0, 1s defining each item by the factor/cluster with the row wise largest absolute loading.

...

Note

Although part of the ICLUST set of programs, this is also more useful for factor or principal components analysis.

Author(s)

William Revelle

References

http://personality-project.org/r/r.ICLUST.html

See Also

ICLUST.graph,ICLUST.cluster, cluster.fit , VSS, factor2cluster

---

ICLUST *ICLUST: Item Cluster Analysis – Hierarchical cluster analysis using psychometric principles*

Description

A common data reduction technique is to cluster cases (subjects). Less common, but particularly useful in psychological research, is to cluster items (variables). This may be thought of as an alternative to factor analysis, based upon a much simpler model. The cluster model is that the correlations between variables reflect that each item loads on at most one cluster, and that items that load on those clusters correlate as a function of their respective loadings on that cluster and items that define different clusters correlate as a function of their respective cluster loadings and the intercluster correlations. Essentially, the cluster model is a Very Simple Structure factor model of complexity one (see VSS).

This function applies the ICLUST algorithm to hierarchically cluster items to form composite scales. Clusters are combined if coefficients alpha and beta will increase in the new cluster.

Alpha, the mean split half correlation, and beta, the worst split half correlation, are estimates of the reliability and general factor saturation of the test. (See also the omega function to estimate McDonald's coeffients ω_h and ω_t)

Usage

```
ICLUST(r.mat, nclusters=0, alpha=3, beta=1, beta.size=4, alpha.size=3,
correct=TRUE,correct.cluster=TRUE, reverse=TRUE, beta.min=.5, output=1, digits=2,labels=NULL,cu
n.iterations = 0,title="ICLUST",plot=TRUE)

#ICLUST(r.mat)    #use all defaults
#ICLUST(r.mat,nclusters =3)    #use all defaults and if possible stop at 3 clusters
#ICLUST(r.mat, output =3)     #long output shows clustering history
#ICLUST(r.mat, n.iterations =3)  #clean up solution by item reassignment
```

Arguments

r.mat	A correlation matrix or data matrix/data.frame. (If r.mat is not square i.e, a correlation matrix, the data are correlated using pairwise deletion.
nclusters	Extract clusters until nclusters remain (default will extract until the other criteria are met or 1 cluster, whichever happens first). See the discussion below for alternative techniques for specifying the number of clusters.

alpha	Apply the increase in alpha criterion (0) never or for (1) the smaller, 2) the average, or 3) the greater of the separate alphas. (default = 3)
beta	Apply the increase in beta criterion (0) never or for (1) the smaller, 2) the average, or 3) the greater of the separate betas. (default =1)
beta.size	Apply the beta criterion after clusters are of beta.size (default = 4)
alpha.size	Apply the alpha criterion after clusters are of size alpha.size (default =3)
correct	Correct correlations for reliability (default = TRUE)
correct.cluster	Correct cluster -sub cluster correlations for reliability of the sub cluster , default is TRUE))
reverse	Reverse negative keyed items (default = TRUE
beta.min	Stop clustering if the beta is not greater than beta.min (default = .5)
output	1) short, 2) medium, 3) long output (default =1)
labels	vector of item content or labels
cut	sort cluster loadings > absolute(cut) (default = 0)
n.iterations	iterate the solution n.iterations times to "purify" the clusters (default = 0)
digits	Precision of digits of output (default = 2)
title	Title for this run
plot	Should ICLUST.rgraph be called automatically for plotting (requires Rgraphviz default=TRUE)

Details

Extensive documentation and justification of the algorithm is available in the original MBR 1979 http://personality-project.org/revelle/publications/iclust.pdf paper. Further discussion of the algorithm and sample output is available on the personality-project.org web page: http://personality-project.org/r/r.ICLUST.html

The results are best visualized using ICLUST.graph, the results of which can be saved as a dot file for the Graphviz program. http://www.graphviz.org/. With the installation of Rgraphviz, ICLUST will automatically provide cluster graphs.

A common problem in the social sciences is to construct scales or composites of items to measure constructs of theoretical interest and practical importance. This process frequently involves administering a battery of items from which those that meet certain criteria are selected. These criteria might be rational, empirical,or factorial. A similar problem is to analyze the adequacy of scales that already have been formed and to decide whether the putative constructs are measured properly. Both of these problems have been discussed in numerous texts, as well as in myriad articles. Proponents of various methods have argued for the importance of face validity, discriminant validity, construct validity, factorial homogeneity, and theoretical importance.

Revelle (1979) proposed that hierachical cluster analysis could be used to estimate a new coefficient (beta) that was an estimate of the general factor saturation of a test. More recently, Zinbarg, Revelle, Yovel and Li (2005) compared McDonald's Omega to Chronbach's

alpha and Revelle's beta. They conclude that ω_h hierarchical is the best estimate. An algorithm for estimating `omega` is available as part of this package.

Revelle and Zinbarg (2009) discuss alpha, beta, and omega, as well as other estimates of reliability.

ICLUST was completely rewritten for the psych package. Please email me if you want help with this version of ICLUST or if you desire more features.

A requested feature (not yet available) is to specify certain items as forming a cluster. That is, to do confirmatory cluster analysis.

The program currently has three primary functions: cluster, loadings, and graphics.

Although ICLUST will give what it thinks is the best solution in terms of the number of clusters to extract, the user will sometimes disagree. To get more clusters than the default solution, just set the nclusters parameter to the number desired. However, to get fewer than meet the alpha and beta criteria, it is sometimes necessary to set alpha=0 and beta=0 and then set the nclusters to the desired number.

Clustering 24 tests of mental ability

A sample output using the 24 variable problem by Harman can be represented both graphically and in terms of the cluster order. Note that the graphic is created using GraphViz in the dot language. `ICLUST.graph` produces the dot code for Graphviz. Somewhat lower resolution graphs with fewer options are available in the `ICLUST.rgraph` function which requires Rgraphviz. Dot code can be viewed directly in Graphviz or can be tweaked using commercial software packages (e.g.,OmniGraffle)

Note that for this problem, with these parameters, the data form one large cluster. (This is consistent with the Very Simple Structure (`VSS`) output as well, which shows a clear one factor solution for complexity 1 data.) See below for an example with this same data set, but with more stringent parameter settings.

At least for the Harman 24 mental ability measures, it is interesting to compare the cluster pattern matrix with the oblique rotation solution from a factor analysis. The factor congruence of a four factor oblique pattern solution with the four cluster solution is $> .99$ for three of the four clusters and $> .97$ for the fourth cluster.

To see the graphic output go to `http://personality-project.org/r/r.ICLUST.html` or use `ICLUST.rgraph` (requires Rgraphviz).

Value

`title`	Name of this run
`results`	A list containing the step by step cluster history, including which pair was grouped, what were the alpha and betas of the two groups and of the combined group. Note that the alpha values are "standardized alphas" based upon the correlation matrix, rather than the raw alphas that will come from `score.items` The print.psych and summary.psych functions will print out just the must important results.
`corrected`	The raw and corrected for alpha reliability cluster intercorrelations.
`clusters`	a matrix of -1,0, and 1 values to define cluster membership.

`purified` A list of the cluster definitions and cluster loadings of the purified solution.

To show just the most salient items, use the cutoff option in `print.psych`

`cluster.fit, structure.fit, pattern.fit`

There are a number of ways to evaluate how well any factor or cluster matrix reproduces the original matrix. Cluster fit considers how well the clusters fit if only correlations with clusters are considered. Structure fit evaluates R = CC' while pattern fit evaluate R = C inverse (phi) C' where C is the cluster loading matrix, and phi is the intercluster correlation matrix.

Author(s)

William Revelle

References

Revelle, W. Hierarchical Cluster Analysis and the Internal Structure of Tests. Multivariate Behavioral Research, 1979, 14, 57-74.

Revelle, W. and Zinbarg, R. E. (2009) Coefficients alpha, beta, omega and the glb: comments on Sijtsma. Psychometrika, 2009.

`http://personality-project.org/revelle/publications/iclust.pdf`
See also more extensive documentation at `http://personality-project.org/r/r.ICLUST.html` and
Revelle, W. (in prep) An introduction to psychometric theory with applications in R. To be published by Springer. (working draft available at `http://personality-project.org/r/book/`

See Also

`ICLUST.graph`,`ICLUST.cluster`, `cluster.fit` , `VSS`, `omega`

Examples

```
test.data <- Harman74.cor$cov
ic.out <- ICLUST(test.data)
summary(ic.out)
ic.out <- ICLUST(test.data,nclusters =4)  #use all defaults and stop at 4 clusters
print(ic.out)
plot(ic.out)    #this shows the spatial representation
```

`interp.median` *Find the interpolated sample median, quartiles, or specific quantiles for a vector, matrix, or data frame*

Description

For data with a limited number of response categories (e.g., attitude items), it is useful treat each response category as range with width, w and linearly interpolate the median, quartiles, or any quantile value within the median response.

Usage

```
interp.median(x, w = 1,na.rm=TRUE)
interp.quantiles(x, q = .5, w = 1,na.rm=TRUE)
interp.quartiles(x,w=1,na.rm=TRUE)
interp.boxplot(x,w=1,na.rm=TRUE)
interp.values(x,w=1,na.rm=TRUE)
interp.qplot.by(y,x,w=1,na.rm=TRUE,xlab="group",ylab="dependent",ylim=NULL,arrow.len=.05,typ="b
```

Arguments

`x`	input vector
`q`	quantile to estimate ($0 < q < 1$
`w`	category width
`y`	input vector for interp.qplot.by
`na.rm`	should missing values be removed
`xlab`	x label
`ylab`	Y label
`ylim`	limits for the y axis
`arrow.len`	length of arrow in interp.qplot.by
`typ`	plot type in interp.qplot.by
`add`	add the plot or not
`...`	additional parameters to plotting function

Details

If the total number of responses is N, with median, M, and the number of responses at the median value, Nm >1, and Nb= the number of responses less than the median, then with the assumption that the responses are distributed uniformly within the category, the interpolated median is M - .5w + w*(N/2 - Nb)/Nm.

The generalization to 1st, 2nd and 3rd quartiles as well as the general quantiles is straightforward.

A somewhat different generalization allows for graphic presentation of the difference between interpolated and non-interpolated points. This uses the interp.values function.

If the input is a matrix or data frame, quantiles are reported for each variable.

Value

im	interpolated median(quantile)
v	interpolated values for all data points

See Also

median

Examples

```
interp.median(c(1,2,3,3,3))  # compare with median = 3
interp.median(c(1,2,2,5))
interp.quantiles(c(1,2,2,5),.25)
x <- sample(10,100,TRUE)
interp.quartiles(x)
#
x <-  c(1,1,2,2,2,3,3,3,3,4,5,1,1,1,2,2,3,3,3,3,4,5,1,1,1,2,2,3,3,3,3,4,2)
y <-  c(1,2,3,3,3,3,4,4,4,5,5,1,2,3,3,3,3,4,4,5,5,5,1,5,3,3,3,3,4,4,4,5,5)
x <-  x[order(x)]   #sort the data by ascending order to make it clearer
y <- y[order(y)]
xv <- interp.values(x)
yv <- interp.values(y)
barplot(x,space=0,xlab="ordinal position",ylab="value")
lines(1:length(x)-.5,xv)
points(c(length(x)/4,length(x)/2,3*length(x)/4),interp.quartiles(x))
barplot(y,space=0,xlab="ordinal position",ylab="value")
lines(1:length(y)-.5,yv)
points(c(length(y)/4,length(y)/2,3*length(y)/4),interp.quartiles(y))
data(galton)
interp.median(galton)
interp.qplot.by(galton$child,galton$parent,ylab="child height"
,xlab="Mid parent height")
```

---

iqitems *14 multiple choice IQ items*

Description

14 multiple choice ability items were included as part of the Synthetic Aperture Personality Assessment (SAPA) web based personality assessment project. The data from 1000 subjects are included here as a demonstration set for scoring multiple choice inventories and doing basic item statistics.

Usage

```
data(iqitems)
```

Format

A data frame with 1000 observations on the following 14 variables.

`iq1` In the following number series, what number comes next?

`iq8` Please mark the word that does not match the other words:

`iq10` If you rearrange the letters ATNHIDLA, you will have the name of a:

`iq15` If Jerks are Perks and some Perks are Lerks, then some Jerks are definitely Lerks. This statement is:

`iq20` How many total legs do two ducks and three dogs have?

`iq44` Matrix reasoning 2

`iq47` Matrix reasoning 5

`iq2` In the following number series, what number comes next? 1 2 4 7 12

`iq11` The opposite of a 'stubborn' person is a ' ' person.

`iq16` Zach is taller than Matt and Richard is shorter than Zach. Which of the following statements would be most accurate?

`iq32` If the day before yesterday is three days after Saturday then what day is today?

`iq37` In the following alphanumeric series, what letter comes next? Q, S, N, P, L

`iq43` Matrix Reasoning 1

`iq49` Matrix Reasoning 9

Details

14 items were sampled from 54 items given as part of the SAPA project (Revelle, Wilt and Rosenthal, 2009) to develop online measures of ability.

Source

`http://personality-project.org`

References

Revelle, William, Wilt, Joshua, and Rosenthal, Allen (2009) Personality and Cognition: The Personality-Cognition Link. In Gruszka, Alexandra and Matthews, Gerald and Szymura, Blazej (Eds.) Handbook of Individual Differences in Cognition: Attention, Memory and Executive Control, Springer.

Examples

```
data(iqitems)
iq.keys <- c(4,4,3,1,4,3,2,3,1,4,1,3,4,3)
score.multiple.choice(iq.keys,iqitems)
```

`irt.item.diff.rasch` *Simple function to estimate item difficulties using IRT concepts*

Description

Steps toward a very crude and preliminary IRT program. These two functions estimate item difficulty and discrimination parameters.

Usage

```
irt.item.diff.rasch(items)
irt.discrim(item.diff,theta,items)
```

Arguments

`items`	a matrix of items
`item.diff`	a vector of item difficulties (found by irt.item.diff)
`theta`	ability estimate from irt.person.theta

Details

Item Response Theory (aka "The new psychometrics") models individual responses to items with a logistic function and an individual (theta) and item difficulty (diff) parameter.

irt.item.diff.rasch finds item difficulties with the assumption of theta=0 for all subjects and that all items are equally discriminating.

irt.discrim takes those difficulties and theta estimates from `irt.person.rasch` to find item discrimination (beta) parameters.

A far better package with these features is the ltm package. The IRT functions in the psych-package are for pedagogical rather than production purposes. They are believed to be accurate, but are not guaranteed. They do seem to be slightly more robust to missing data structures associated with SAPA data sets than the ltm package.

Value

a vector of item difficulties or item discriminations.

Note

Under development. Not recommended for public consumption.

Author(s)

William Revelle

See Also

`irt.person.rasch`

`irt.1p` *Item Response Theory estimate of theta (ability) using a Rasch (like) model*

Description

Item Response Theory models individual responses to items by estimating individual ability (theta) and item difficulty (diff) parameters. This is an early and crude attempt to capture this modeling procedure.

Usage

```
irt.person.rasch(diff, items)
irt.0p(items,possible=20)
irt.1p(delta,items)
irt.2p(delta,beta,items)
```

Arguments

`diff`	A vector of item difficulties –probably taken from irt.item.diff.rasch
`items`	A matrix of 0,1 items nrows = number of subjects, ncols = number of items
`possible`	Number of items in the scale – used to determine values of all wrong or all right
`delta`	delta is the same as diff and is the item difficulty parameter
`beta`	beta is the item discrimination parameter found in `irt.discrim`

Details

A very preliminary IRT estimation procedure. Given scores xij for ith individual on jth item
Classical Test Theory ignores item difficulty and defines ability as expected score : abilityi = theta(i) = x(i.) A zero parameter model rescales these mean scores from 0 to 1 to a quasi logistic scale ranging from - 4 to 4 This is merely a non-linear transform of the raw data to reflect a logistic mapping.

Basic 1 parameter (Rasch) model considers item difficulties (delta j): p(correct on item j for the ith subject |theta i, deltaj) = 1/(1+exp(deltaj - thetai)) If we have estimates of item difficulty (delta), then we can find theta i by optimization

Two parameter model adds item sensitivity (beta j): p(correct on item j for subject i |thetai, deltaj, betaj) = 1/(1+exp(betaj *(deltaj- theta i))) Estimate delta, beta, and theta to maximize fit of model to data.

The procedure used here is to first find the item difficulties assuming theta = 0 Then find theta given those deltas Then find beta given delta and theta.

This is not an "official" way to do IRT, but is useful for basic item development.

Value

a data.frame with estimated ability (theta) and quality of fit. (for irt.person.rasch)
a data.frame with the raw means, theta0, and the number of items completed

Note

Not recommended for serious use. This code is under development.

Author(s)

William Revelle

See Also

`irt.item.diff.rasch`

`wkappa` *Find Cohen's kappa and weighted kappa coefficients for correlation of two raters*

Description

Cohen's kappa (Cohen, 1960) and weighted kappa (Cohen, 1968) may be used to find the agreement of two raters when using nominal scores.

wkappa is (probability of observed matches - probability of expected matches)/(1 - probability of expected matches). Kappa just considers the matches on the main diagonal. Weighted kappa considers off diagonal elements as well

Usage

```
wkappa(x, w = NULL)
```

Arguments

`x`	Either a two by n data with categorical values from 1 to p or a p x p table. If a data array, a table will be found.
`w`	A p x p matrix of weights. If not specified, they are set to be 1 (on the diagonal) and .5$\hat{}$(distance from diagonal) off the diagonal.

Details

Some categorical judgments are made using more than two outcomes. For example, two diagnosticians might be asked to categorize patients three ways (e.g., Personality disorder, Neurosis, Psychosis). Just as base rates affect observed cell frequencies in a two by two table, they need to be considered in the n-way table (Cohen, 1960).

A more useful measure of the agreement between two raters when the data are quantitative is the Intra Class Correlation (`ICC`).

Value

`kappa` Unweighted kappa

`weighted.kappa`
If weights are provided

Note

kappa is included in psych more for completeness than necessity. The Kappa function in the vcd package is probably preferred.

To avoid confusion with Kappa (from vcd) or the kappa function from base, the function is named wkappa

Author(s)

William Revelle

References

Cohen, J. (1960). A coefficient of agreement for nominal scales. Educational and Psychological Measurement, 20 37-46

Cohen, J. (1968). Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. Psychological Bulletin, 70, 213-220.

Examples

```
cohen <- matrix(c(
0.44, 0.05, 0.01,
0.07, 0.20, 0.03,
0.09, 0.05, 0.06),ncol=3)

wkappa(cohen)

fleiss <- matrix(c(
 0.53, 0.05, 0.02,
 0.11, 0.14, 0.05,
 0.01, 0.06, 0.03),ncol=3)

weights <- matrix(c(
 1.0000, 0.0000, 0.4444,
 0.0000, 1.0000, 0.6666,
 0.4444, 0.6666, 1.0000),ncol=3)

 wkappa(fleiss,weights)
```

make.keys *Create a keys matrix for use by score.items or cluster.cor*

Description

When scoring items by forming composite scales either from the raw data using `score.items` or from the correlatio matrix using `cluster.cor`, it is necessary to create a keys matrix. This is just a short cut for doing so.

Usage

```
make.keys(nvars, keys.list, key.labels = NULL, item.labels = NULL)
```

Arguments

`nvars`	Number of variables items to be scored
`keys.list`	A list of the scoring keys,one element for each scale
`key.labels`	Labels for the scales can be specified here, or in the key.list
`item.labels`	Typically, just the colnames of the items data matrix.

Details

There are two ways to create keys for the `score.items` function. One is to laboriously do it in a spreadsheet and then copy them into R. The other is to just specify them by item number in a list.

Value

`keys`	a nvars x nkeys matrix of -1, 0, or 1s describing how to score each scale. nkeys is the length of the keys.list

See Also

`score.items`, `cluster.cor`,

Examples

```
data(attitude)
 key.list <- list(all=c(1,2,3,4,-5,6,7),
                  first=c(1,2,3),
                  last=c(4,5,6,7))
 keys <- make.keys(7,key.list,item.labels = colnames(attitude))
 keys

 scores <- score.items(keys,attitude,short=TRUE)
 scores

 data(bfi)
```

```
keys.list <- list(agree=c(-1,2:5),conscientious=c(6:8,-9,-10),extraversion=c(-11,-12,13:15),neuroticism=
keys <- make.keys(25,keys.list,item.labels=colnames(bfi))
scores <- score.items(keys,bfi,short=TRUE)
scores
```

`mat.regress` *Multiple Regression from matrix input*

Description

Extract subsets of variables (x and y) from a correlation or data matrix matrix and find the multiple correlation and beta weights of the (x) set predicting each member of the (y) set.

Usage

```
mat.regress(m, x, y,n.obs=NULL,digits=2)
```

Arguments

`m`	a matrix of correlations or, if not square of data
`x`	the column numbers of the x set (e.g., c(1,3,5)
`y`	the column numbers of the y set (e.g., c(2,4,6)
`n.obs`	If specified, then confidence intervals, etc. are calculated, not needed if raw data are given
`digits`	round the answer to digits

Details

Although it is more common to calculate multiple regression from raw data, it is, of course, possible to do so from a set of correlations. The input to the function is a square covariance or correlation matrix, as well as the column numbers of the x (predictor) and y (criterion) variables. The program will find correlations if given raw data.

The output is a set of multiple correlations, one for each dependent variable in the y set.

A typical use in the SAPA project is to form item composites by clustering or factoring (see `ICLUST`, `principal`), extract the clusters from these results (`factor2cluster`), and then form the composite correlation matrix using `cluster.cor`. The variables in this reduced matrix may then be used in multiple R procedures using mat.regress.

Although the overall matrix can have missing correlations, the correlations in the subset of the matrix used for prediction must exist.

If the number of observations is entered, then the conventional confidence intervals, statistical significance, and shrinkage estimates are reported.

If the input matrix is rectangular, correlations are found from the data.

Value

beta	the beta weights for each variable in X for each variable in Y
R	The multiple R for each equation (the amount of change a unit in the predictor set leads to in the criterion set).
R2	The multiple R2 (% variance acounted for) for each equation

Author(s)

William Revelle

Maintainer: William Revelle <revelle@northwestern.edu>

See Also

cluster.cor, factor2cluster,principal,ICLUST

Examples

```
## Not run:
test.data <- Harman74.cor$cov     #24 mental variables
#choose 3 of them to regress against another 4 -- arbitrary choice of variables
print(mat.regress(test.data,c(1,2,3),c(4,5,10,12)),digits=2)
## End(Not run)
#gives this output
#print(mat.regress(test.data,c(1,2,3),c(4,5,10,12)),digits=2)
#$beta
#                 Flags GeneralInformation Addition CountingDots
#VisualPerception  0.40               0.22     0.16         0.30
#Cubes             0.06               0.18     0.06         0.05
#PaperFormBoard    0.12               0.10    -0.16         0.00
#
#$R
#             Flags GeneralInformation           Addition       CountingDots
#             0.49               0.38              0.18               0.32
#
#$R2
#             Flags GeneralInformation           Addition       CountingDots
#              0.24               0.15              0.03               0.10
#
#
data(attitude)
mat.regress(attitude,c(1:3),c(4:7)) #standardized regression from raw data
```

`matrix.addition` *A function to add two vectors or matrices*

Description

It is sometimes convenient to add two vectors or matrices in an operation analogous to matrix multiplication. For matrices nXm and mYp, the matrix sum of the i,jth element of nSp = sum(over m) of iXm + mYj.

Usage

```
x %+% y
```

Arguments

x	a n by m matrix (or vector if m= 1)
y	a m by p matrix (or vector if m = 1)

Details

Used in such problems as Thurstonian scaling. Although not technically matrix addition, as pointed out by Krus, there are many applications where the sum or difference of two vectors or matrices is a useful operation. An alternative operation for vectors is outer(x ,y , FUN="+") but this does not work for matrices.

Value

a n by p matix of sums

Author(s)

William Revelle

References

Krus, D. J. (2001) Matrix addition. Journal of Visual Statistics, 1, (February, 2001).

Examples

```
x <- seq(1,4)
z <- x %+% -t(x)
x
z
#compare with outer(x,-x,FUN="+")
x <- matrix(seq(1,6),ncol=2)
y <- matrix(seq(1,10),nrow=2)
z <- x %+% y
x
```

```
y
z
#but compare this with outer(x ,y,FUN="+")
```

`multi.hist` *Multiple histograms with density and normal fits on one page*

Description

Given a matrix or data.frame, produce histograms for each variable in a "matrix" form. Include normal fits and density distributions for each plot.

The number of rows and columns may be specified, or calculated.

May be used for single variables.

Usage

```
multi.hist(x,nrow=NULL, ncol=NULL,density=TRUE,main="Histogram, Density, and Normal Fit")
```

Arguments

`x`	matrix or data.frame
`nrow`	number of rows in the plot
`ncol`	number of columns in the plot
`density`	density=TRUE, show the normal fits and density distributions
`main`	title for each panel

Author(s)

William Revelle

Examples

```
#multi.hist(attitude[-1])
```

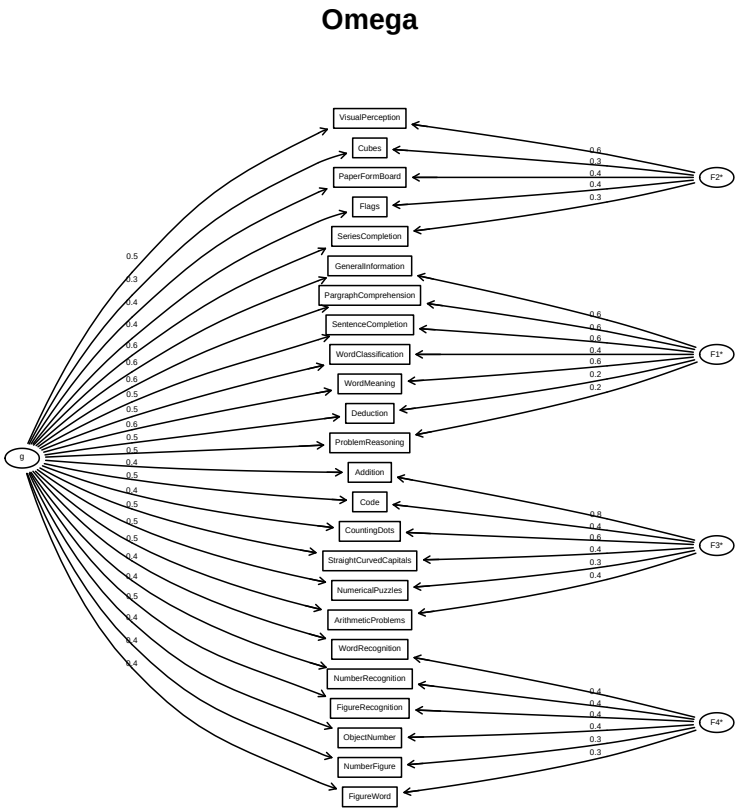

Figure 6: Hierarchical factor solutions are typical in the ability domain where a g factor is thought to reflect the correlations among lower level factors. An alternative transformation is to ortogonalize the g factor from the residual group factors using the Schmid-Leiman transformaton (Figure 7)

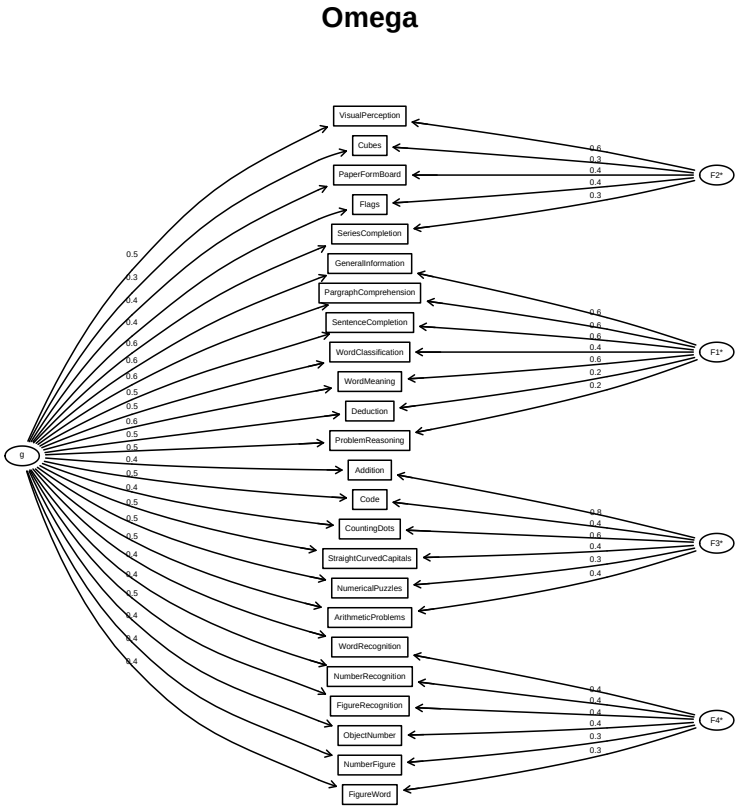

Figure 7: An alternative to the standard hierarchical factor solutions which are typical in the ability domain is to orthgonalize the g factor from the residual group factors using the Schmid-Leiman transformaton. For the hierarchical solution, see Figure 6

`omega.graph` *Graph hierarchical factor structures*

Description

Hierarchical factor structures represent the correlations between variables in terms of a smaller set of correlated factors which themselves can be represented by a higher order factor.

Two alternative solutions to such structures are found by the `omega` function. The correlated factors solutions represents the effect of the higher level, general factor, through its effect on the correlated factors. The other representation makes use of the Schmid Leiman transformation to find the direct effect of the general factor upon the original variables as well as the effect of orthogonal residual group factors upon the items.

Graphic presentations of these two alternatives are helpful in understanding the structure. omega.graph draws both such structures. Graphs are drawn directly onto the graphics window or expressed in "dot" commands for conversion to graphics using implementations of Graphviz.

Using Graphviz allows the user to clean up the Rgraphviz output.

In addition

Usage

```
omega.graph(om.results, out.file = NULL,  sl = TRUE, labels = NULL, size = c(8, 6), node.font =
```

Arguments

`om.results`	The output from the omega function
`out.file`	Optional output file for off line analysis using Graphviz
`sl`	Orthogonal clusters using the Schmid-Leiman transform (sl=TRUE) or oblique clusters
`labels`	variable labels
`size`	size of graphics window
`node.font`	What font to use for the items
`edge.font`	What font to use for the edge labels
`rank.direction`	Defaults to left to right
`digits`	Precision of labels
`title`	Figure title
`...`	Other options to pass into the graphics packages

Details

Requires the Rgraphviz package. omega requires the GPArotation package.

Value

clust.graph	A graph object
sem	A matrix suitable to be run throughe the sem function in the sem package.

Note

Requires rgraphviz. – omega requires GPArotation

Author(s)

http://personality-project.org/revelle.html
Maintainer: William Revelle ⟨ revelle@northwestern.edu ⟩

References

http://personality-project.org/r/r.omega.html

Revelle, W. (in preparation) An Introduction to Psychometric Theory with applications in R. http://personality-project.org/r/book

Revelle, W. (1979). Hierarchical cluster analysis and the internal structure of tests. Multivariate Behavioral Research, 14, 57-74. (http://personality-project.org/revelle/publications/iclust.pdf)

Zinbarg, R.E., Revelle, W., Yovel, I., & Li. W. (2005). Cronbach's Alpha, Revelle's Beta, McDonald's Omega: Their relations with each and two alternative conceptualizations of reliability. Psychometrika. 70, 123-133. http://personality-project.org/revelle/publications/zinbarg.revelle.pmet.05.pdf

Zinbarg, R., Yovel, I., Revelle, W. & McDonald, R. (2006). Estimating generalizability to a universe of indicators that all have one attribute in common: A comparison of estimators for omega. Applied Psychological Measurement, 30, 121-144. DOI: 10.1177/0146621605278814 http://apm.sagepub.com/cgi/reprint/30/2/121

See Also

omega, make.hierarchical, ICLUST.rgraph

Examples

```
#24 mental tests from Holzinger-Swineford-Harman
if(require(GPArotation) ) {om24 <- omega(Harman74.cor$cov,4) } #run omega
if(require(Rgraphviz) ){om24pn <- omega.graph(om24,sl=FALSE)} #show the structure
#
#example hierarchical structure from Jensen and Weng
if(require(GPArotation) ) {jen.omega <- omega(make.hierarchical())}
if(require(Rgraphviz) ) {om.jen <- omega.graph(jen.omega,sl=FALSE) }
```

`omega` *Calculate the omega estimate of factor saturation*

Description

McDonald has proposed coefficient omega as an estimate of the general factor saturation of a test. One way to find omega is to do a factor analysis of the original data set, rotate the factors obliquely, do a Schmid Leiman transformation, and then find omega. This function estimates omega as suggested by McDonald by using hierarchical factor analysis (following Jensen).

Usage

```
omega(m, nfactors=3, pc = "mle",key = NULL, flip=TRUE, digits=2,title="Omega",sl=TRUE,labels=NU
```

Arguments

`m`	A correlation matrix or a data.frame/matrix of data
`nfactors`	Number of factors believed to be group factors
`pc`	pc="pa" for principal axes, pc="pc" for principal components, pc="mle" for maximum likelihood.
`key`	a vector of +/- 1s to specify the direction of scoring of items. The default is to assume all items are positively keyed, but if some items are reversed scored, then key should be specified.
`flip`	If flip is TRUE, then items are automatically flipped to have positive correlations on the general factor. Items that have been reversed are shown with a - sign.
`digits`	if specified, round the output to digits
`title`	Title for this analysis
`sl`	If plotting the results, should the Schmid Leiman solution be shown or should the hierarchical solution be shown? (default sl=TRUE)
`labels`	If plotting, what labels should be applied to the variables? If not specified, will default to the column names.
`plot`	plot=TRUE (default) calls omega.graph, plot =FALSE does not
`n.obs`	Number of observations - used for goodness of fit statistic
`rotate`	What rotation to apply? The default is oblimin, the alternative is simplimax.
`...`	Allows additional parameters to be passed through to the factor routines

Details

"Many scales are assumed by their developers and users to be primarily a measure of one latent variable. When it is also assumed that the scale conforms to the effect indicator model of measurement (as is almost always the case in psychological assessment), it is important to support such an interpretation with evidence regarding the internal structure of that scale. In particular, it is important to examine two related properties pertaining to the internal structure of such a scale. The first property relates to whether all the indicators forming the scale measure a latent variable in common.

The second internal structural property pertains to the proportion of variance in the scale scores (derived from summing or averaging the indicators) accounted for by this latent variable that is common to all the indicators (Cronbach, 1951; McDonald, 1999; Revelle, 1979). That is, if an effect indicator scale is primarily a measure of one latent variable common to all the indicators forming the scale, then that latent variable should account for the majority of the variance in the scale scores. Put differently, this variance ratio provides important information about the sampling fluctuations when estimating individuals' standing on a latent variable common to all the indicators arising from the sampling of indicators (i.e., when dealing with either Type 2 or Type 12 sampling, to use the terminology of Lord, 1956). That is, this variance proportion can be interpreted as the square of the correlation between the scale score and the latent variable common to all the indicators in the infinite universe of indicators of which the scale indicators are a subset. Put yet another way, this variance ratio is important both as reliability and a validity coefficient. This is a reliability issue as the larger this variance ratio is, the more accurately one can predict an individual's relative standing on the latent variable common to all the scale's indicators based on his or her observed scale score. At the same time, this variance ratio also bears on the construct validity of the scale given that construct validity encompasses the internal structure of a scale." (Zinbarg, Yovel, Revelle, and McDonald, 2006).

McDonald has proposed coefficient omega (hierarchical (ω_h) as an estimate of the general factor saturation of a test. Zinbarg, Revelle, Yovel and Li (2005) `http://personality-project.org/revelle/publications/zinbarg.revelle.pmet.05.pdf` compare McDonald's ω_h to Cronbach's α and Revelle's β They conclude that ω_h is the best estimate. (See also Zinbarg et al., 2006 and Revelle and Zinbarg (2009)).

One way to find ω_h is to do a factor analysis of the original data set, rotate the factors obliquely, factor that correlation matrix, do a Schmid-Leiman (schmid) transformation to find general factor loadings, and then find ω_h. Here we present code to do that.

ω_h differs as a function of how the factors are estimated. Three options are available, pc="pa" does a principle axes factor analysis (factor.pa), pc="mle" uses the factanal function, and pc="pc" does a principal components analysis (principal).

For ability items, it is typically the case that all items will have positive loadings on the general factor. However, for non-cognitive items it is frequently the case that some items are to be scored positively, and some negatively. Although probably better to specify which directions the items are to be scored by specifying a key vector, if flip =TRUE (the default), items will be reversed so that they have positive loadings on the general factor. The keys are reported so that scores can be found using the `score.items` function.

Output from omega will be shown graphically using the `omega.graph` function. This requires Rgraphviz to be installed. If Rgraphviz is not available, select plot=FALSE.

β, an alternative to ω, is defined as the worst split half reliability. It can be estimated by using ICLUST (a hierarchical clustering algorithm originally developed for main frames and written in Fortran and that is now available in R. (For a very complimentary review of why the ICLUST algorithm is useful in scale construction, see Cooksey and Soutar, 2005).

The `omega` function uses exploratory factor analysis to estimate the ω_h coefficient. It is important to remember that "A recommendation that should be heeded, regardless of the method chosen to estimate ω_h, is to always examine the pattern of the estimated general factor loadings prior to estimating ω_h. Such an examination constitutes an informal test of the assumption that there is a latent variable common to all of the scale's indicators that can be conducted even in the context of EFA. If the loadings were salient for only a relatively small subset of the indicators, this would suggest that there is no true general factor underlying the covariance matrix. Just such an informal assumption test would have afforded a great deal of protection against the possibility of misinterpreting the misleading ω_h estimates occasionally produced in the simulations reported here." (Zinbarg et al., 2006, p 137).

A simple demonstration of the problem of an omega estimate reflecting just one of two group factors can be found in the last example.

Although omega is uniquely defined only for cases where 3 or more subfactors are extracted, it is sometimes desired to have a two factor solution. This is done by forcing the schmid extraction to treat the two subfactors as having equal loadings. See Zinbarg et al., 2007.

In addition to ω_h, another of McDonald's coefficients is ω_t. This is an estimate of the total reliability of a test.

McDonald's ω_t, which is similar to Guttman's λ_6, `guttman` but uses the estimates of uniqueness (u^2 from factor analysis to find e_j^2. This is based on a decomposition of the variance of a test score, V_x into four parts: that due to a general factor, $\vec{g}$, that due to a set of group factors, $\vec{f}$, (factors common to some but not all of the items), specific factors, $\vec{s}$ unique to each item, and $\vec{e}$, random error. (Because specific variance can not be distinguished from random error unless the test is given at least twice, some combine these both into error).

Letting $\vec{x} = c\vec{g} + \vec{A}\vec{f} + \vec{D}\vec{s} + \vec{e}$ then the communality of item$_j$, based upon general as well as group factors, $h_j^2 = c_j^2 + \sum f_{ij}^2$ and the unique variance for the item $u_j^2 = \sigma_j^2(1 - h_j^2)$ may be used to estimate the test reliability. That is, if h_j^2 is the communality of item$_j$, based upon general as well as group factors, then for standardized items, $e_j^2 = 1 - h_j^2$ and

$$\omega_t = \frac{\vec{1}c\vec{c}'\vec{1} + \vec{1}\vec{A}\vec{A}'\vec{1}'}{V_x} = 1 - \frac{\sum(1 - h_j^2)}{V_x} = 1 - \frac{\sum u^2}{V_x}$$

Because $h_j^2 \geq r_{smc}^2$, $\omega_t \geq \lambda_6$.

It is important to distinguish here between the two ω coefficients of McDonald, 1978 and Equation 6.20a of McDonald, 1999, ω_t and ω_h. While the former is based upon the sum of squared loadings on all the factors, the latter is based upon the sum of the squared loadings on the general factor.

$$\omega_h = \frac{\vec{1}c\vec{c}'\vec{1}}{V_x}$$

Value

`omega hierarchical`

The ω_h coefficient

`omega total`	The $omega_t$ coefficient
`alpha`	Cronbach's α
`schmid`	The Schmid Leiman transformed factor matrix and associated matrices
`schmid$sl`	The g factor loadings as well as the residualized factors
`schmid$orthog`	Varimax rotated solution of the original factors
`schmid$oblique`	The oblimin or promax transformed factors
`schmid$fcor`	the correlation matrix of the oblique factors
`schmid$gloading`	The loadings on the higher order, g, factor of the oblimin factors
`key`	A vector of -1 or 1 showing which direction the items were scored.
`model`	a matrix suitable to be given to the sem function for structure equation models

Note

Requires the GPArotation package.

The default rotation uses oblimin from the GPArotation package. Alternatives include the simplimax function, as well as promax.

Author(s)

`http://personality-project.org/revelle.html`
Maintainer: William Revelle ⟨ revelle@northwestern.edu ⟩

References

`http://personality-project.org/r/r.omega.html`

Revelle, W. and Zinbarg, R. E. (2009) Coefficients alpha, beta, omega and the glb: comments on Sijtsma. Psychometrika (in press).

Revelle, W. (1979). Hierarchical cluster analysis and the internal structure of tests. Multivariate Behavioral Research, 14, 57-74. (`http://personality-project.org/revelle/publications/iclust.pdf`)

Zinbarg, R.E., Revelle, W., Yovel, I., & Li. W. (2005). Cronbach's Alpha, Revelle's Beta, McDonald's Omega: Their relations with each and two alternative conceptualizations of reliability. Psychometrika. 70, 123-133. `http://personality-project.org/revelle/publications/zinbarg.revelle.pmet.05.pdf`

Zinbarg, R., Yovel, I. & Revelle, W. (2007). Estimating omega for structures containing two group factors: Perils and prospects. Applied Psychological Measurement. 31 (2), 135-157.

Zinbarg, R., Yovel, I., Revelle, W. & McDonald, R. (2006). Estimating generalizability to a universe of indicators that all have one attribute in common: A comparison of estimators for omega. Applied Psychological Measurement, 30, 121-144. DOI: 10.1177/0146621605278814 `http://apm.sagepub.com/cgi/reprint/30/2/121`

See Also

omega.graph ICLUST, ICLUST.graph, VSS, schmid , make.hierarchical

Examples

```
## Not run:
test.data <- Harman74.cor$cov
my.omega <- omega(test.data,3)
print(my.omega,digits=2)
#

## End(Not run)
#create 9 variables with a hierarchical structure
jen.data <- sim.hierarchical()
#with correlations of
jen.data
#find omega
if(require(Rgraphviz)) {jen.omega <- omega(jen.data,digits=2)} else {jen.omega <- omega(jen.data,digits=2
jen.omega

#create 8 items with a two factor solution, showing the use of the flip option
#sim2 <- item.sim(8)
#omega(sim2)   #an example of misidentification-- remember to look at the loadings matrices.
#apply omega to analyze 6 mental ability tests
data(ability.cov)   #has a covariance matrix
if(require(Rgraphviz)) {omega(ability.cov$cov)} else {omega(ability.cov$cov,plot=FALSE)}
```

---

p.rep *Find the probability of replication for an F, t, or r and estimate effect size*

Description

The probability of replication of an experimental or correlational finding as discussed by Peter Killeen (2005) is the probability of finding an effect in the same direction upon an exact replication. For articles submitted to Psychological Science, p.rep needs to be reported.

F, t, p and r are all estimates of the size of an effect. But F, t, and p also are also a function of the sample size. Effect size, d prime, may be expressed as differences between means compared to within cell standard deviations, or as a correlation coefficient. These functions convert p, F, and t to d prime and the r equivalent.

Usage

```
p.rep(p = 0.05, n=NULL,twotailed = FALSE)
p.rep.f(F,df2,twotailed=FALSE)
p.rep.r(r,n,twotailed=TRUE)
p.rep.t(t,df,df2=NULL,twotailed=TRUE)
```

Arguments

p	conventional probability of statistic (e.g., of F, t, or r)
F	The F statistic
df	Degrees of freedom of the t-test, or of the first group if unequal sizes
df2	Degrees of freedom of the denominator of F or the second group in an unequal sizes t test
r	Correlation coefficient
n	Total sample size if using r
t	t-statistic if doing a t-test or testing significance of a regression slope
twotailed	Should a one or two tailed test be used?

Details

The conventional Null Hypothesis Significance Test (NHST) is the likelihood of observing the data given the null hypothesis of no effect. But this tells us nothing about the probability of the null hypothesis. Peter Killeen (2005) introduced the probability of replication as a more useful measure. The probability of replication is the probability that an exact replication study will find a result in the *same direction* as the original result.

p.rep is based upon a 1 tailed probability value of the observed statistic.

Other frequently called for statistics are estimates of the effect size, expressed either as Cohen's d, Hedges g, or the equivalent value of the correlation, r.

For p.rep.t, if the cell sizes are unequal, the effect size estimates are adjusted by the ratio of the mean cell size to the harmonic mean cell size (see Rownow et al., 2000).

Value

p.rep	Probability of replication
dprime	Effect size (Cohen's d) if more than just p is specified
prob	Probability of F, t, or r. Note that this can be either the one-tailed or two tailed probability value.
r.equivalent	For t-tests, the r equivalent to the t (see Rosenthal and Rubin(2003), Rosnow, Rosenthal, and Rubin, 2000))

.

Note

The p.rep value is the one tailed probability value of obtaining a result in the same direction.

References

Cummings, Geoff (2005) Understanding the average probability of replication: comment on Killeen 2005). Psychological Science, 16, 12, 1002-1004).

Killeen, Peter H. (2005) An alternative to Null-Hypothesis Significance Tests. Psychological Science, 16, 345-353

Rosenthal, R. and Rubin, Donald B.(2003), r-sub(equivalent): A Simple Effect Size Indicator. Psychological Methods, 8, 492-496.

Rosnow, Ralph L., Rosenthal, Robert and Rubin, Donald B. (2000) Contrasts and correlations in effect-size estimation, Psychological Science, 11. 446-453.

Examples

```
p.rep(.05)  #probability of replicating a result if the original study had a  p = .05
p.rep.f(9.0,98)  #probability of replicating  a result with F = 9.0 with 98 df
p.rep.r(.4,50)    #probability of replicating a result if r =.4 with n = 50
p.rep.t(3,98)   #probability of replicating a result if t = 3 with df =98
p.rep.t(2.14,84,14) #effect of equal sample sizes (see Rosnow et al.)
```

---

paired.r *Test the difference between (un)paired correlations*

Description

Test the difference between two (paired or unpaired) correlations. Given 3 variables, x, y, z, is the correlation between xy different than that between xz? If y and z are independent, this is a simple t-test of the z transformed rs. But, if they are dependent, it is a bit more complicated.

Usage

```
paired.r(xy, xz, yz=NULL, n, n2=NULL,twotailed=TRUE)
```

Arguments

`xy`	r(xy)
`xz`	r(xz)
`yz`	r(yz)
`n`	Number of subjects for first group
`n2`	Number of subjects in second group (if not equal to n)
`twotailed`	Calculate two or one tailed probability values

Details

To find the z of the difference between two independent correlations, first convert them to z scores using the Fisher r-z transform and then find the z of the difference between the two correlations. The default assumption is that the group sizes are the same, but the test can be done for different size groups by specifying n2.

If the correlations are not independent (i.e., they are from the same sample) then the correlation with the third variable r(yz) must be specified. Find a t statistic for the difference of thee two dependent correlations.

Value

a list containing the calculated t or z values and the associated two (or one) tailed probability.

`t`	t test of the difference between two dependent correlations
`p`	probability of the t or of the z
`z`	z test of the difference between two independent correlations

Author(s)

William Revelle

See Also

`p.rep.r`, `cor.test`

Examples

```
paired.r(.5,.3, .4, 100) #dependent correlations
paired.r(.5,.3,NULL,100) #independent correlations same sample size
paired.r(.5,.3,NULL, 100, 64) # independent correlations, different sample sizes
```

`pairs.panels` *SPLOM, histograms and correlations for a data matrix*

Description

Adapted from the help page for pairs, pairs.panels shows a scatter plot of matrices (SPLOM), with bivariate scatter plots below the diagonal, histograms on the diagonal, and the Pearson correlation above the diagonal. Useful for descriptive statistics of small data sets. If lm=TRUE, linear regression fits are shown for both y by x and x by y. Correlation ellipses are also shown.

Usage

```
pairs.panels(x, smooth = TRUE, scale = FALSE, density=TRUE,ellipses=TRUE,digits = 2, pch = 20,l
```

Arguments

`x`	a data.frame or matrix
`smooth`	TRUE draws loess smooths
`scale`	TRUE scales the correlation font by the size of the absolute correlation.
`density`	TRUE shows the density plots as well as histograms
`ellipses`	TRUE draws correlation ellipses
`lm`	Plot the linear fit rather than the LOESS smoothed fits.
`digits`	the number of digits to show
`pch`	The plot character (defaults to 20 which is a '.').
`jiggle`	Should the points be jittered before plotting?
`...`	other options for pairs

Details

Shamelessly adapted from the pairs help page. Uses panel.cor, panel.cor.scale, and panel.hist, all taken from the help pages for pairs. Also adapts the ellipse function from John Fox's car package.

`pairs.panels` is most useful when the number of variables to plot is less than about 6-8. It is particularly useful for an initial overview of the data.

Value

A scatter plot matrix (SPLOM) is drawn in the graphic window. The lower off diagonal draws scatter plots, the diagonal histograms, the upper off diagonal reports the Pearson correlation (with pairwise deletion).

If lm=TRUE, then the scatter plots are drawn above and below the diagonal, each with a linear regression fit. Useful to show the difference between regression lines.

See Also

`pairs`

Examples

```
pairs.panels(attitude)   #see the graphics window
data(peas)
pairs.panels(peas,lm=TRUE,xlim=c(14,22),ylim=c(14,22))
```

`partial.r` *Find the partial correlations for a set (x) of variables with set (y) removed.*

Description

A straightforward application of matrix algebra to remove the effect of the variables in the y set from the x set. Input may be either a data matrix or a correlation matrix. Variables in x and y are specified by location.

Usage

```
partial.r(m, x, y, digits = 2)
```

Arguments

`m`	A data or correlation matrix
`x`	The variable numbers associated with the X set.
`y`	The variable numbers associated with the Y set
`digits`	Report correlations to digits of accuracy (default =2)

Details

It is sometimes convenient to partial the effect of a number of variables (e.g., sex, age, education) out of the correlations of another set of variables. This could be done laboriously by finding the residuals of various multiple correlations, and then correlating these residuals. The matrix algebra alternative is to do it directly.

Value

The matrix of partial correlations.

Author(s)

William Revelle

References

Revelle, W. (in prep) An introduction to psychometric theory with applications in R. To be published by Springer. (working draft available at `http://personality-project.org/r/book/`

See Also

`mat.regress` for a similar application for regression

Examples

```
jen <- make.hierarchical()    #make up a correlation matrix
round(jen[1:5,1:5],2)
par.r <- partial.r(jen,c(1,3,5),c(2,4))
par.r
```

`peas` *Galton's Peas*

Description

Francis Galton introduced the correlation coefficient with an analysis of the similarities of the parent and child generation of 700 sweet peas.

Usage

```
data(peas)
```

Format

A data frame with 700 observations on the following 2 variables.

`parent` The mean diameter of the mother pea for 700 peas

`child` The mean diameter of the daughter pea for 700 sweet peas

Details

Galton's introduction of the correlation coefficient was perhaps the most important contribution to the study of individual differences. This data set allows a graphical analysis of the data set. There are two different graphic examples. One shows the regression lines for both relationships, the other finds the correlation as well.

Source

Stanton, Jeffrey M. (2001) Galton, Pearson, and the Peas: A brief history of linear regression for statistics intstructors, Journal of Statistics Education, 9. (retrieved from the web from http://www.amstat.org/publications/jse/v9n3/stanton.html) reproduces the table from Galton, 1894, Table 2.

The data were generated from this table.

References

Galton, Francis (1877) Typical laws of heredity. paper presented to the weekly evening meeting of the Royal Institution, London. Volume VIII (66) is the first reference to this data set. The data appear in

Galton, Francis (1894) Natural Inheritance (5th Edition), New York: MacMillan).

Examples

```
data(peas)
pairs.panels(peas,lm=TRUE,xlim=c(14,22),ylim=c(14,22))
describe(peas)
pairs.panels(peas)
```

---

phi.demo *Create demo data for psychometrics*

Description

A not very interesting demo of what happens if bivariate continuous data are dichotomized. Bascially a demo of r, phi, and polychor.

Usage

```
phi.demo(n=1000,r=.6, cuts=c(-2,-1,0,1,2))
```

Arguments

n	number of cases to simulate
r	correlation between latent and observed
cuts	form dichotomized variables at the value of cuts

Details

A demonstration of the problem of different base rates on the phi correlation, and how these are partially solved by using the polychoric correlation. Not one of my more interesting demonstrations. See http://personality-project.org/r/simulating-personality.html and http://personality-project.org/r/r.datageneration.html for better demonstrations of data generation.

Value

a matrix of correlations and a graphic plot)

Author(s)

William Revelle

References

http://personality-project.org/r/simulating-personality.html and http://personality-project.org/r/r.datageneration.html for better demonstrations of data generation.

See Also

VSS.simulate,item.sim

Examples

```
round(phi.demo() ,2)  #compare the phi (lower off diagonal and polychoric correlations (upper off diagona
```

`phi` *Find the phi coefficient of correlation between two dichotomous variables*

Description

Given a 1 x 4 vector or a 2 x 2 matrix of frequencies, find the phi coefficient of correlation. Typical use is in the case of predicting a dichotomous criterion from a dichotomous predictor.

Usage

```
phi(t, digits = 2)
```

Arguments

`t`	a 1 x 4 vector or a 2 x 2 matrix
`digits`	round the result to digits

Details

In many prediction situations, a dichotomous predictor (accept/reject) is validated against a dichotomous criterion (success/failure). Although a polychoric correlation estimates the underlying Pearson correlation as if the predictor and criteria were continuous and bivariate normal variables, the phi coefficient is the Pearson applied to a matrix of 0's and 1s.

Given a two x two table of counts

a	b	a+b
c	d	c+d
a+c	b+d	a+b+c+d

convert all counts to fractions of the total and then \ Phi = a- (a+b)*(a+c)/sqrt((a+b)(c+d)(a+c)(b+d))

Value

phi coefficient of correlation

Author(s)

William Revelle with modifications by Leo Gurtler

See Also

`phi2poly` ,`Yule`, `Yule2phi`

Examples

```
phi(c(30,20,20,30))
phi(c(40,10,10,40))
x <- matrix(c(40,5,20,20),ncol=2)
phi(x)
```

`phi2poly` *Convert a phi coefficient to a polychoric correlation*

Description

Given a phi coefficient (a Pearson r calculated on two dichotomous variables), and the marginal frequencies (in percentages), what is the corresponding estimate of the polychoric correlation?

Given a two x two table of counts

a	b
c	d

The phi coefficient is (a - (a+b)*(a+c))/sqrt((a+b)(a+c)(b+d)(c+c)).

This function reproduces the cell entries for specified marginals and then calls John Fox's polychor function.

Usage

```
phi2poly(ph, cp, cc)
```

Arguments

`ph`	phi
`cp`	probability of the predictor – the so called selection ratio
`cc`	probability of the criterion – the so called success rate.

Details

Uses John Fox's polycor package, which in turn requires the mvtnorm package

Value

a polychoric correlation

Author(s)

William Revelle

See Also

`polychor.matrix`, `Yule2phi.matrix`, `phi2poly.matrix`

Examples

```
#phi2poly(.3,.5,.5)
#phi2poly(.3,.3,.7)
```

`plot.psych` *Plotting functions for the psych package of class "psych"*

Description

Combines several plotting functions into one for objects of class "psych". This can be used to plot the results of `VSS`, `ICLUST`, `omega`, `factor.pa`, or `principal`.

Usage

```
plot.psych(x,labels=NULL,...)
```

Arguments

`x`	The object to plot
`labels`	Variable labels
`...`	other calls to plot

Details

Passes the appropriate values to plot

Value

Graphic output for factor analysis and cluster analysis.

Note

More precise plotting control is available in the separate plot functions.

Author(s)

William Revelle

See Also

`VSS.plot` and `cluster.plot`

Examples

```
test.data <- Harman74.cor$cov
f4 <- factor.pa(test.data,4)
plot(f4)
```

`polar` *Convert Cartesian factor loadings into polar coordinates*

Description

Factor and cluster analysis output typically presents item by factor correlations (loadings). Tables of factor loadings are frequently sorted by the size of loadings. This style of presentation tends to make it difficult to notice the pattern of loadings on other, secondary, dimensions. By converting to polar coordinates, it is easier to see the pattern of the secondary loadings.

Usage

```
polar(f, sort = TRUE)
```

Arguments

`f`	A matrix of loadings or the output from a factor or cluster analysis program
`sort`	sort=TRUE: sort items by the angle of the items on the first pair of factors.

Details

Although many uses of factor analysis/cluster analysis assume a simple structure where items have one and only one large loading, some domains such as personality or affect items have a more complex structure and some items have high loadings on two factors. (These items are said to have complexity 2, see `VSS`). By expressing the factor loadings in polar coordinates, this structure is more readily perceived.

For each pair of factors, item loadings are converted to an angle with the first factor, and a vector length corresponding to the amount of variance in the item shared with the two factors.

For a two dimensional structure, this will lead to a column of angles and a column of vector lengths. For n factors, this leads to n* (n-1)/2 columns of angles and an equivalent number of vector lengths.

Value

`polar`	A data frame of polar coordinates

Author(s)

William Revelle

References

Rafaeli, E. & Revelle, W. (2006). A premature consensus: Are happiness and sadness truly opposite affects? Motivation and Emotion. \

Hofstee, W. K. B., de Raad, B., & Goldberg, L. R. (1992). Integration of the big five and circumplex approaches to trait structure. Journal of Personality and Social Psychology, 63, 146-163.

See Also

`ICLUST`, `cluster.plot`, `circ.tests`, `factor.pa`

Examples

```
circ.data <- circ.sim(24,500)
circ.fa <- factor.pa(circ.data,2)
circ.polar <- round(polar(circ.fa),2)
circ.polar
#compare to the graphic
cluster.plot(circ.fa)
```

---

poly.mat *Find polychoric correlations of item data*

Description

Uses John Fox's hetcor function (from polychor package) to find a matrix of polychoric correlations for integer data. Essentially a wrapper for hetcor to convert integer item data into factor (categorical) data and then use hetcor. Just a useful shortcut for subsequent factor analysis.

Usage

```
poly.mat(x, short = TRUE, std.err = FALSE, ML = FALSE)
```

Arguments

`x`	A matrix or data frame of integer data
`short`	short=TRUE, just show the correlations, short=FALSE give the full hetcor output
`std.err`	std.err=FALSE does not report the standard errors (faster)
`ML`	ML=FALSE do a quick two step procedure, ML=TRUE, do longer maximum likelihood

Details

Typical personality and item data are integer values (0,1 for ability; 1,2, 3, 4 for attitude scales). The normal correlation procedures will find Pearson correlations (cor). The polycor and hetcor functions from John Fox's polychor package will find polychoric correlations for categorical data. This wrapper function converts integer data to categorical data and then calls hetcor.

Value

A matrix of polychoric correlations (if short=TRUE), otherwise a list of various estimates (see hetcor).

Note

requires polycor

Author(s)

William Revelle

Examples

```
round(phi.demo() ,2)  #compare the phi (lower off diagonal and polychoric correlations (upper off diagona
```

---

polychor.matrix *Phi or Yule coefficient matrix to polychoric coefficient matrix*

Description

Given a matrix of phi or Yule correlation coefficients and a vector of marginals, use John Fox's polycor function to convert these to polychoric correlations.

Some older correlation matrices were reported as matrices of Phi or of Yule correlations. That is, correlations were found from the two by two table of counts:

a	b
c	d

Yule Q is (ad - bc)/(ad+bc).

With marginal frequencies of a+b, c+d, a+c, b+d.

Given a square matrix of such correlations, and the proportions for each variable that are in the a + b cells, it is possible to reconvert each correlation into a two by two table and then estimate the corresponding polychoric correlation (using John Fox's polychor function.

Usage

```
Yule2poly.matrix(x, v)
phi2poly.matrix(x, v)
Yule2phi.matrix(x, v)
```

Arguments

x	a matrix of phi or Yule coefficients
v	A vector of marginal frequencies

Details

These functions call `Yule2poly`, `Yule2phi` or `phi2poly` for each cell of the matrix. See those functions for more details. See `phi.demo` for an example.

Value

A matrix of correlations

Author(s)

William Revelle

Examples

```
round(phi.demo() ,2)  #compare the phi (lower off diagonal and polychoric correlations (upper off diagona
```

`principal` *Principal components analysis*

Description

Does an eigen value decomposition and returns eigen values, loadings, and degree of fit for a specified number of components. Basically it is just doing a principal components for n principal components. Can show the residual correlations as well. The quality of reduction in the squared correlations is reported by comparing residual correlations to original correlations. Unlike princomp, this returns a subset of just the best nfactors. The eigen vectors are rescaled by the sqrt of the eigen values to produce the component loadings more typical in factor analysis.

Usage

```
principal(r, nfactors = 1, residuals = FALSE,rotate="varimax",n.obs=NA, scores=FALSE,missing=FA
```

Arguments

r	a correlation matrix. If a raw data matrix is used, the correlations will be found using pairwise deletions for missing values.
nfactors	Number of components to extract
residuals	FALSE, do not show residuals, TRUE, report residuals
rotate	"none", "varimax", "promax" or "oblimin" are possible rotations of the solution.
n.obs	Number of observations used to find the correlation matrix if using a correlation matrix. Used for finding the goodness of fit statistics.
scores	If TRUE, estimate component scores
missing	if scores are TRUE, and missing=TRUE, then impute missing values using either the median or the mean
impute	"median" or "mean" values are used to replace missing values
digits	digits =2. Accuracy of answers as well as display

Details

Useful for those cases where the correlation matrix is improper (perhaps because of SAPA techniques).

There are a number of data reduction techniques including principal components and factor analysis. Both PC and FA attempt to approximate a given correlation or covariance matrix of rank n with matrix of lower rank (p). ${}_nR_n \approx_n F_{kk}F'_n + U^2$ where k is much less than n. For principal components, the item uniqueness is assumed to be zero and all elements of the correlation matrix are fitted. That is, ${}_nR_n \approx_n F_{kk}F'_n$ The primary empirical difference between a components versus a factor model is the treatment of the variances for each item. Philosophically, components are weighted composites of observed variables while in the factor model, variables are weighted composites of the factors.

For a n x n correlation matrix, the n principal components completely reproduce the correlation matrix. However, if just the first k principal components are extracted, this is the best k dimensional approximation of the matrix.

It is important to recognize that rotated principal components are not principal components (the axes associated with the eigen value decomposition) but are merely components.

Some of the statistics reported are more appropriate for maximum likelihood factor analysis rather than principal components analysis, and are reported to allow comparisons with these other models.

Although for items, it is typical to find component scores by scoring the salient items (using, e.g.,`score.items` component scores can be estimated by regression. This is done just to be parallel with the principal axis factor analysis function `factor.pa`

Value

values	Eigen Values of all components – useful for a scree plot
rotation	which rotation was requested?
n.obs	number of observations specified or found

communality	Communality estimates for each item. These are merely the sum of squared factor loadings for that item.
loadings	A standard loading matrix of class "loadings"
fit	Fit of the model to the correlation matrix
fit.off	how well are the off diagonal elements reproduced?
residual	Residual matrix – if requested
communality	The history of the communality estimates. Probably only useful for teaching what happens in the process of iterative fitting.
dof	Degrees of Freedom for this model. This is the number of observed correlations minus the number of independent parameters (number of items * number of factors - nf*(nf-1)/2. That is, dof = niI * (ni-1)/2 - ni * nf + nf*(nf-1)/2.
objective	value of the function that is minimized by maximum likelihood procedures. This is reported for comparison purposes and as a way to estimate chi square goodness of fit. The objective function is $f = log(trace((FF' + U2)^{-1}R) - log(\|(FF' + U2)^{-1}R\|) - n.items.$
STATISTIC	If the number of observations is specified or found, this is a chi square based upon the objective function, f. Using the formula from factanal: $\chi^2 = (n.obs - 1 - (2 * p + 5)/6 - (2 * factors)/3)) * f$
PVAL	If n.obs > 0, then what is the probability of observing a chisquare this large or larger?
phi	If oblique rotations (using oblimin from the GPArotation package) are requested, what is the interfactor correlation.
scores	If scores=TRUE, then estimates of the factor scores are reported

Author(s)

William Revelle

References

Revelle, W. An introduction to psychometric theory with applications in R (in prep) Springer. Draft chapters available at http://personality-project.org/r/book/

See Also

VSS,factor2cluster,factor.pa, factor.congruence

Examples

```
#Four principal components of the Harmon 24 variable problem
#compare to a four factor principal axes solution using factor.congruence
pc <- principal(Harman74.cor$cov,4,rotate="varimax")
pa <- factor.pa(Harman74.cor$cov,4,rotate="varimax")
round(factor.congruence(pc,pa),2)
```

print.psych *Print and summary functions for the psych class*

Description

Give limited output (print) or somewhat more detailed (summary) for the factor.pa, omega, ICLUST, score.items, cluster.cor, cluster.loadings and the sim functions. In addition, will supply the factor correlations for output from a promax rotation applied to a factanal output.

Usage

```
print.psych(x,digits=2,all=FALSE,cutoff=NULL,sort=FALSE,...)
summary.psych(object,digits=2,items=FALSE,...)
```

Arguments

x	Output from a psych function (e.g., factor.pa, omega,ICLUST, score.items, cluster.cor
object	Output from a psych function
items	items=TRUE (default) does not print the item whole correlations
digits	Number of digits to use in printing
all	if all=TRUE, then the object is declassed and all output from the function is printed
cutoff	Cluster loadings < cutoff will not be printed. For factor.pa, cutoff defaults to .3, for omega to .2.
sort	Cluster loadings are in sorted order
...	More options to pass to summary and print

Details

Most of the psych functions produce too much output. print.psych and summary.psych use generic methods for printing just the highlights. To see what else is available, either ask for the structure (str(theobject).

To get complete output, unclass(theobject) and then print it.

As an added feature, if the promax function is applied to a factanal loadings matrix, the normal output just provides the rotation matrix. print.psych will provide the factor correlations. (Following a suggestion by John Fox and Uli Keller to the R-help list). The alternative is to just use the Promax function directly on the factanal object.

Value

Various psych functions produce copious output. This is a way to summarize the most important parts of the output of the score.items, cluster.scores, and ICLUST functions. See those (score.items, cluster.cor, cluster.loadings, or ICLUST) for details on what is produced.

Note

See `score.items`, `cluster.cor`, `cluster.loadings`, or `ICLUST`for details on what is printed.

Author(s)

William Revelle

Examples

```
data(bfi)
 keys.list <- list(agree=c(-1,2:5),conscientious=c(6:8,-9,-10),extraversion=c(-11,-12,13:15),neuroticism=
 keys <- make.keys(25,keys.list,item.labels=colnames(bfi))
 scores <- score.items(keys,bfi,short=TRUE)
 scores
 summary(scores)
```

`Promax` *Perform promax rotation and return the inter factor angles*

Description

promax is an oblique rotation function introduced by Hendrickson and White (1964) and implemented in the promax function in the stats package. Unfortunately, promax does not report the inter factor correlations. Promax does.

Usage

```
Promax(x, m = 4)
```

Arguments

`x`	A loadings matrix
`m`	the power to which to raise the varimax loadings

Details

This is a very direct adaptation of the stats::promax function. The addition is that it will return the interfactor correlations as well as the loadings and rotation matrix.

In addition, it will take output from either the factanal, factor.pa, or principal functions and select just the loadings matrix for analysis.

Value

`loadings`	Oblique factor loadings
`rotmat`	The rotation matrix applied to the original loadings to produce the promax soluion
`Phi`	The interfactor correlation matrix

Note

A direct adaptation of the stats:promax function following suggestions to the R-help list by Ulrich Keller and John Fox.

Author(s)

William Revelle

References

Hendrickson, A. E. and White, P. O, 1964, British Journal of Statistical Psychology, 17, 65-70.

See Also

`promax`, `factor.pa`

Examples

```
jen <- sim.hierarchical()
f3 <- factor.pa(jen,3)
Promax(f3)
m3 <- factanal(covmat=jen,factors=3)
Promax(m3)
```

---

r.test *Tests of significance for correlations*

Description

Tests the significance of a single correlation, the difference between two independent correlations, the difference between two dependent correlations sharing one variable, or the difference between two dependent correlations with different variables.

Usage

```
r.test(n, r12, r34 = NULL, r23 = NULL, r13 = NULL, r14 = NULL, r24 = NULL, n2 = NULL,pooled=TRU
```

Arguments

`n`	Sample size of first group
`r12`	Correlation to be tested
`r34`	Test if this correlation is different from r12, if r23 is specified, but r13 is not, then r34 becomes r13
`r23`	if ra = r(12) and rb = r(13) then test for differences of dependent correlations given r23

r13	implies ra =r(12) and rb =r(34) test for difference of dependent correlations
r14	implies ra =r(12) and rb =r(34)
r24	ra =r(12) and rb =r(34)
n2	n2 is specified in the case of two independent correlations. n2 defaults to n if if not specified
pooled	use pooled estimates of correlations
twotailed	should a twotailed or one tailed test be used

Details

Depending upon the input, one of four different tests of correlations is done.\ 1) For a sample size n, find the t value for a single correlation. \ 2) For sample sizes of n and n2 (n2 = n if not specified) find the z of the difference between the z transformed correlations divided by the standard error of the difference of two z scores. \ 3) For sample size n, and correlations ra= r12, rb= r23 and r13 specified, test for the difference of two dependent correlations. \ 4) For sample size n, test for the difference between two dependent correlations involving different variables.\ For clarity, correlations may be specified by value. If specified by location and if doing the test of dependent correlations, if three correlations are specified, they are assumed to be in the order r12, r13, r23.

Value

test	Label of test done
z	z value for tests 2 or 4
t	t value for tests 1 and 3
p	probability value of z or t

Note

Steiger specifically rejects using the Hotelling T test to test the difference between correlated correlations. These tests follow Steiger's advice.

Author(s)

William Revelle

References

Olkin, I. and Finn, J. D. (1995). Correlations redux. Psychological Bulletin, 118(1):155-164. \ Steiger, J.H. (1980), Tests for comparing elements of a correlation matrix, Psychological Bulletin, 87, 245-251.

See Also

This extends the tests in paired.r,r.con

Examples

```
n <- 30
r <- seq(0,.9,.1)
rc <- matrix(r.con(r,n),ncol=2)
test <- r.test(n,r)
r.rc <- data.frame(r=r,z=fisherz(r),lower=rc[,1],upper=rc[,2],t=test$t,p=test$p)
round(r.rc,2)

r.test(50,r)
r.test(30,.4,.6)       #test the difference between two independent correlations
r.test(103,.4,.5,.1)   #Steiger case A
r.test(103,.5,.6,.7,.5,.5,.8)  #steiger Case B
```

---

read.clipboard *shortcut for reading from the clipboard*

Description

input from the keyboard is easy but a bit obscure, particularly for Mac users. This is just an easier mnemonic to do so. Also will do some processing to read lower triangular matrices and fill them out to square matrices.

Usage

```
read.clipboard(header = TRUE, ...)
read.clipboard.lower(diag=TRUE,names=NULL)
read.clipboard.upper(diag=TRUE,names=NULL)
#my.data <- read.clipboad()         #assumes headers and tab delimited
#my.data <- read.clipboard.csv()     #assumes heades and comma delimited
```

Arguments

header	Does the first row have variable labels
diag	for upper or lower triangular matrices, is the diagonal specified or not
names	for read.clipboard.lower or upper, what colnames to use
...	Other parameters to pass to read

Details

A typical session of R might involve data stored in text files, generated on line, etc. Although it is easy to just read from a file (particularly if using file.locate(), copying from the file to the clipboard and then reading from the clipboard is also very convenient (and somewhat more intuitive to the naive user. This is particularly convenient when copying from a text book or article and just moving a section of text into R.)

Based upon a suggestion by Ken Knoblauch to the R-help listserve.

read.clipboard.lower and read.clipboard.upper are adapted from John Fox's read.moments function in the sem package. They will read a lower (or upper) triangular matrix from the clipboard and return a full, symmetric matrix for use by factanal, `factor.pa` , `ICLUST` , etc. If the diagonal is false, it will be replaced by 1.0s. These two function were added to allow easy reading of examples from various texts and manuscripts with just triangular output.

Value

the contents of the clipboard.

Author(s)

William Revelle

Examples

```
#my.data <- read.clipboad()
#my.data <- read.clipboard.csv()
#my.data <- read.clipboad(header=FALSE)
```

---

rescale *Function to convert scores to "conventional " metrics*

Description

Psychologists frequently report data in terms of transformed scales such as "IQ" (mean=100, sd=15, "SAT/GRE" (mean=500, sd=100), "ACT" (mean=18, sd=6), "T-scores" (mean=50, sd=10), or "Stanines" (mean=5, sd=2). The `rescale` function converts the data to standard scores and then rescales to the specified mean(s) and standard deviation(s).

Usage

```
rescale(x, mean = 100, sd = 15,df=TRUE)
```

Arguments

`x`	A matrix or data frame
`mean`	Desired mean of the rescaled scores- may be a vector
`sd`	Desired standard deviation of the rescaled scores
`df`	if TRUE, returns a data frame, otherwise a matrix

Value

A data.frame (default) or matrix of rescaled scores.

Author(s)

William Revelle

See Also

See Also `scale`

Examples

```
T <- rescale(attitude,50,10) #all put on same scale
describe(T)
T1 <- rescale(attitude,seq(0,300,50),seq(10,70,10)) #different means and sigmas
describe(T1)
```

`sat.act` *3 Measures of ability: SATV, SATQ, ACT*

Description

Self reported scores on the SAT Verbal, SAT Quantitative and ACT were collected as part of the Synthetic Aperture Personality Assessment (SAPA) web based personality assessment project. Age, gender, and education are also reported. The data from 700 subjects are included here as a demonstration set for correlation and analysis.

Usage

```
data(sat.act)
```

Format

A data frame with 700 observations on the following 6 variables.

`gender` males = 1, females = 2

`education` self reported education 1 = high school ... 5 = graduate work

`age` age

`ACT` ACT composite scores may range from 1 - 36. National norms have a mean of 20.

`SATV` SAT Verbal scores may range from 200 - 800.

`SATQ` SAT Quantitative scores may range from 200 - 800

Details

hese items were collected as part of the SAPA project to develop online measures of ability (Revelle, Wilt and Rosenthal, 2009). The score means are higher than national norms suggesting both self selection for people taking on line personality and ability tests and a self reporting bias in scores.

See also the iq.items data set.

Source

`http://personality-project.org`

References

Revelle, William, Wilt, Joshua, and Rosenthal, Allen (2009) Personality and Cognition: The Personality-Cognition Link. In Gruszka, Alexandra and Matthews, Gerald and Szymura, Blazej (Eds.) Handbook of Individual Differences in Cognition: Attention, Memory and Executive Control, Springer.

Examples

```
data(sat.act)
describe(sat.act)
pairs.panels(sat.act)
```

---

scaling.fits *Test the adequacy of simple choice, logistic, or Thurstonian scaling.*

Description

Given a matrix of choices and a vector of scale values, how well do the scale values capture the choices? That is, what is size of the squared residuals given the model versus the size of the squared choice values?

Usage

```
scaling.fits(model, data, test = "logit", digits = 2, rowwise = TRUE)
```

Arguments

model	A vector of scale values
data	A matrix or dataframe of choice frequencies
test	"choice", "logistic", "normal"
digits	Precision of answer
rowwise	Are the choices ordered by column over row (TRUE) or row over column False)

Details

How well does a model fit the data is the classic problem of all of statistics. One fit statistic for scaling is the just the size of the residual matrix compared to the original estimates.

Value

GF	Goodness of fit of the model
original	Sum of squares for original data
resid	Sum of squares for residuals given the data and the model
residual	Residual matrix

Note

Mainly for demonstration purposes for a course on psychometrics

Author(s)

William Revelle

References

Revelle, W. (in preparation) Introduction to psychometric theory with applications in R, Springer. http://personality-project.org/r/book

See Also

thurstone, vegetables

---

schmid *Apply the Schmid Leiman transformation to a correlation matrix*

Description

One way to find omega is to do a factor analysis of the original data set, rotate the factors obliquely, do a Schmid Leiman transformation, and then find omega. Here is the code for Schmid Leiman. The S-L transform takes a factor or PC solution, transforms it to an oblique solution, factors the oblique solution to find a higher order (g) factor, and then residualizes g out of the the group factors.

Usage

```
schmid(model, nfactors = 3, pc = "pa",digits=2,rotate="oblimin",n.obs=NA,...)
```

Arguments

model	A correlation matrix
nfactors	Number of factors to extract
pc	pc="pa" for principal axes, pc="pc" for principal components, pc="mle" for maximum likelihood
digits	if digits not equal NULL, rounds to digits
rotate	The default, oblimin, produces somewhat more correlated factors than the alternative, simplimax. The third option is the promax criterion
n.obs	Number of observations, used to find fit statistics if specified. Will be calculated if input is raw data
...	Allows additional parameters to be passed to the factoring routines

Details

Schmid Leiman orthogonalizations are typical in the ability domain, but are not seen as often in the non-cognitive personality domain. S-L is one way of finding the loadings of items on the general factor for estimating omega.

A typical example would be in the study of anxiety and depression. A general neuroticism factor (g) accounts for much of the variance, but smaller group factors of tense anxiety, panic disorder, depression, etc. also need to be considerd.

An alternative model is to consider hierarchical cluster analysis techniques such as `ICLUST`.

Requires the GPArotation package.

Although 3 factors are the minimum number necessary to define the solution uniquely, it is occasionally useful to allow for a two factor solution. This is done here by setting the general factor loadings between the two lower order factors as the sqrt(oblique correlations between the factors). A warning message is issued.

Value

`sl`	loadings on g + nfactors group factors, communalities, uniqueness
`orthog`	original orthogonal factor loadings
`oblique`	oblique factor loadings
`phi`	correlations among the transformed factors
`gload`	loadings of the lower order factors on g
...	

Author(s)

William Revelle

References

`http://personality-project.org/r/r.omega.html` gives an example taken from Jensen and Weng, 1994 of a S-L transformation.

See Also

`omega`, `omega.graph`, `fa.graph`, `ICLUST`,`VSS`

Examples

```
jen <- sim.hierarchical()  #create a hierarchical demo
o.jen <- schmid(jen,digits=2)  #use the oblimin rotation
p.jen <- schmid(jen,rotate="promax") #use the promax rotation
```

`score.alpha` *Score scales and find Cronbach's alpha as well as associated statistics*

Description

Given a matrix or data.frame of k keys for m items (-1, 0, 1), and a matrix or data.frame of items scores for m items and n people, find the sum scores or average scores for each person and each scale. In addition, report Cronbach's alpha, the average r, the scale intercorrelations, and the item by scale correlations. (Superseded by `score.items`).

Usage

```
score.alpha(keys, items, labels = NULL, totals=TRUE,digits = 2)
```

Arguments

`keys`	A matrix or dataframe of -1, 0, or 1 weights for each item on each scale
`items`	Data frame or matrix of raw item scores
`labels`	column names for the resulting scales
`totals`	Find sum scores (default) or average score
`digits`	Number of digits for answer (default =2)

Details

The process of finding sum or average scores for a set of scales given a larger set of items is a typical problem in psychometric research. Although the structure of scales can be determined from the item intercorrelations, to find scale means, variances, and do further analyses, it is typical to find the sum or the average scale score.

Various estimates of scale reliability include "Cronbach's alpha", and the average interitem correlation. For k = number of items in a scale, and av.r = average correlation between items in the scale, alpha = k * av.r/(1+ (k-1)*av.r). Thus, alpha is an increasing function of test length as well as the test homeogeneity.

Alpha is a poor estimate of the general factor saturation of a test (see Zinbarg et al., 2005) for it can seriously overestimate the size of a general factor, and a better but not perfect estimate of total test reliability because it underestimates total reliability. None the less, it is a useful statistic to report.

Value

`scores`	Sum or average scores for each subject on the k scales
`alpha`	Cronbach's coefficient alpha. A simple (but non-optimal) measure of the internal consistency of a test. See also beta and omega.
`av.r`	The average correlation within a scale, also known as alpha 1 is a useful index of the internal consistency of a domain.

n.items	Number of items on each scale
cor	The intercorrelation of all the scales
item.cor	The correlation of each item with each scale. Because this is not corrected for item overlap, it will overestimate the amount that an item correlates with the other items in a scale.

Author(s)

William Revelle

References

An introduction to psychometric theory with applications in R (in preparation). http://personality-project.org/r/book

See Also

score.items, alpha.scale, correct.cor, alpha.scale, cluster.loadings, omega

Examples

```
y <- attitude     #from the datasets package
keys <- matrix(c(rep(1,7),rep(1,4),rep(0,7),rep(-1,3)),ncol=3)
labels <- c("first","second","third")
x <- score.alpha(keys,y,labels)
```

---

score.items — *Score item composite scales and find Cronbach's alpha as well as associated statistics*

Description

Given a matrix or data.frame of k keys for m items (-1, 0, 1), and a matrix or data.frame of items scores for m items and n people, find the sum scores or average scores for each person and each scale. In addition, report Cronbach's alpha, the average r, the scale intercorrelations, and the item by scale correlations. Replace missing values with the item median or mean if desired. Will adjust scores for reverse scored items. See make.keys for a convenient way to make the keys file.

Usage

```
score.items(keys, items, totals = FALSE, ilabels = NULL, missing = TRUE,impute="median",  min =
```

Arguments

keys	A matrix or dataframe of -1, 0, or 1 weights for each item on each scale. May be created by hand, or by using `make.keys`
items	Matrix or dataframe of raw item scores
totals	if TRUE find total scores, if FALSE (default), find average scores
ilabels	a vector of item labels.
missing	TRUE: Replace missing values with the corresponding item median or mean. FALSE: do not score that subject
impute	impute="median" replaces missing values with the item median, impute = "mean" replaces values with the mean response.
min	May be specified as minimum item score allowed, else will be calculated from data
max	May be specified as maximum item score allowed, else will be calculated from data
digits	Number of digits to report
short	if short is TRUE, then just give the item and scale statistics and do not report the scores

Details

The process of finding sum or average scores for a set of scales given a larger set of items is a typical problem in psychometric research. Although the structure of scales can be determined from the item intercorrelations, to find scale means, variances, and do further analyses, it is typical to find scores based upon the sum or the average item score. For some strange reason, personality scale scores are typically given as totals, but attitude scores as averages. The default for score.items is the average.

Various estimates of scale reliability include "Cronbach's alpha", and the average interitem correlation. For k = number of items in a scale, and av.r = average correlation between items in the scale, alpha = k * av.r/(1+ (k-1)*av.r). Thus, alpha is an increasing function of test length as well as the test homeogeneity.

Alpha is a poor estimate of the general factor saturation of a test (see Zinbarg et al., 2005) for it can seriously overestimate the size of a general factor, and a better but not perfect estimate of total test reliability because it underestimates total reliability. None the less, it is a useful statistic to report. To estimate the omega cofficient, use the `omega` function.

Correlations between scales are attenuated by a lack of reliability. Correcting correlations for reliability (by dividing by the square roots of the reliabilities of each scale) sometimes help show structure.

By default, missing values are replaced with the corresponding median value for that item. Means can be used instead (impute="mean"), or subjects with missing data can just be dropped (missing = FALSE).

Value

scores	Sum or average scores for each subject on the k scales

alpha	Cronbach's coefficient alpha. A simple (but non-optimal) measure of the internal consistency of a test. See also beta and omega. Set to 1 for scales of length 1.
av.r	The average correlation within a scale, also known as alpha 1 is a useful index of the internal consistency of a domain. Set to 1 for scales with 1 item.
n.items	Number of items on each scale
item.cor	The correlation of each item with each scale. Because this is not corrected for item overlap, it will overestimate the amount that an item correlates with the other items in a scale.
cor	The intercorrelation of all the scales
corrected	The correlations of all scales (below the diagonal), alpha on the diagonal, and the unattenuated correlations (above the diagonal)

Author(s)

William Revelle

References

An introduction to psychometric theory with applications in R (in preparation). http://personality-project.org/r/book

See Also

make.keys for a convenient way to create the keys file, score.multiple.choice for multiple choice items,
alpha.scale, correct.cor, cluster.cor , cluster.loadings, omega for item/scale analysis

Examples

```
#see  the example including the bfi data set
data(bfi)
 keys.list <- list(agree=c(-1,2:5),conscientious=c(6:8,-9,-10),extraversion=c(-11,-12,13:15),neuroticism=
 keys <- make.keys(25,keys.list,item.labels=colnames(bfi))
 scores <- score.items(keys,bfi,short=TRUE)
 scores
```

`score.multiple.choice` *Score multiple choice items and provide basic test statistics*

Description

Ability tests are typically multiple choice with one right answer. score.multiple.choice takes a scoring key and a data matrix (or data.frame) and finds total or average number right for each participant. Basic test statistics (alpha, average r, item means, item-whole correlations) are also reported.

Usage

```
score.multiple.choice(key, data, score = TRUE, totals = FALSE, ilabels = NULL, missing = TRUE,
```

Arguments

`key`	A vector of the correct item alternatives
`data`	a matrix or data frame of items to be scored.
`score`	score=FALSE, just convert to right (1) or wrong (0). score=TRUE, find the totals or average scores and do item analysis
`totals`	total=FALSE: find the average number correct total=TRUE: find the total number correct
`ilabels`	item labels
`missing`	missing=TRUE: missing values are replaced with means or medians mising=FALSE missing values are not scored
`impute`	impute="median", replace missing items with the median score impute="mean": replace missing values with the item mean
`digits`	How many digits of output
`short`	short=TRUE, just report the item statistics, short=FALSE, report item statistics and subject scores as well

Details

Basically combines `score.items` with a conversion from multiple choice to right/wrong.

The item-whole correlation is inflated because of item overlap.

Value

`scores`	Subject scores on one scale
`missing`	Number of missing items for each subject
`item.stats`	scoring key, response frequencies, item whole correlations, n subjects scored, mean, sd, skew, kurtosis and se for each item
`alpha`	Cronbach's coefficient alpha
`av.r`	Average interitem correlation

Author(s)

William Revelle

See Also

score.items, omega

Examples

```
data(iqitems)
iq.keys <- c(4,4,3,1,4,3,2,3,1,4,1,3,4,3)
score.multiple.choice(iq.keys,iqitems)
#just convert the items to true or false
iq.tf <- score.multiple.choice(iq.keys,iqitems,score=FALSE)
describe(iq.tf)  #compare to previous results
```

SD — *Find the Standard deviation for a vector, matrix, or data.frame - do not return error if there are no cases*

Description

Find the standard deviation of a vector, matrix, or data.frame. In the latter two cases, return the sd of each column. Unlike the sd function, return NA if there are no observations rather than throw an error.

Usage

```
SD(x, na.rm = TRUE)
```

Arguments

x	a vector, data.frame, or matrix
na.rm	na.rm is assumed to be TRUE

Details

Finds the standard deviation of a vector, matrix, or data.frame. Returns NA if no cases.

Just an adaptation of the stats:sd function to return the functionality found in R $<$ 2.7.0 or R $>=$ 2.8.0

Value

The standard deviation

Note

Until R 2.7.0, sd would return a NA rather than an error if no cases were observed. SD brings back that functionality. Although unusual, this condition will arise when analyzing data with high rates of missing values. This function will probably be removed as 2.7.0 becomes outdated.

Author(s)

William Revelle

See Also

These functions use SD rather than sd: `describe.by`, `skew`, `kurtosi`

Examples

```
data(attitude)
sd(attitude) #all complete
attitude[,1] <- NA
SD(attitude) #missing a column
describe(attitude)
```

---

sim.congeneric *Simulate a congeneric data set*

Description

Classical Test Theory (CTT) considers four or more tests to be congenerically equivalent if all tests may be expressed in terms of one factor and a residual error. Parallel tests are the special case where (usually two) tests have equal factor loadings. Tau equivalent tests have equal factor loadings but may have unequal errors. Congeneric tests may differ in both factor loading and error variances.

Usage

```
sim.congeneric(loads = c(0.8, 0.7, 0.6, 0.5),N = NULL,  err=NULL, short = TRUE)
```

Arguments

`N`	How many subjects to simulate. If NULL, return the population model
`loads`	A vector of factor loadings for the tests
`err`	A vector of error variances – if NULL then error = 1 - loading 2
`short`	short=TRUE: Just give the test correlations, short=FALSE, report observed test scores as well as the implied pattern matrix

Details

When constructing examples for reliability analysis, it is convenient to simulate congeneric data structures. These are the most simple of item structures, having just one factor. Mainly used for a discussion of reliability theory as well as factor score estimates.

The implied covariance matrix is just pattern %*% t(pattern).

Value

`model`	The implied population correlation matrix if N=NULL or short=FALSE, otherwise the sample correlation matrix
`pattern`	The pattern matrix implied by the loadings and error variances
`r`	The sample correlation matrix for long output
`observed`	a matrix of test scores for n tests
`latent`	The latent trait and error scores

Author(s)

William Revelle

References

Revelle, W. (in prep) An introduction to psychometric theory with applications in R. To be published by Springer. (working draft available at `http://personality-project.org/r/book/`

See Also

`item.sim` for other simulations, `factor.pa` for an example of factor scores.

Examples

```
test <- sim.congeneric(c(.9,.8,.7,.6))   #just the population matrix
test <- sim.congeneric(c(.9,.8,.7,.6),N=100)   # a sample correlation matrix
test <- sim.congeneric(short=FALSE, N=100)
round(cor(test$observed),2) # show  a congeneric correlation matrix
f1=factor.pa(test$observed,1,scores=TRUE)
round(cor(f1$scores,test$latent),2)  #factor score estimates are correlated with but not equal to the fac
```

`sim.hierarchical` *Create a population or sample correlation matrix, perhaps with hierarchical structure.*

Description

Create a population orthogonal or hierarchical correlation matrix from a set of factor loadings and factor intercorrelations. Samples of size n may be then be drawn from this population. Return either the sample data, sample correlations, or population correlations. This is used to create sample data sets for instruction and demonstration.

Usage

```
sim.hierarchical(gload=NULL, fload=NULL, n = 0, raw = FALSE)
make.hierarchical(gload=NULL, fload=NULL, n = 0, raw = FALSE)  #deprecated
```

Arguments

`gload`	Loadings of group factors on a general factor
`fload`	Loadings of items on the group factor
`n`	Number of subjects to generate: N=0 => population values
`raw`	raw=TRUE, report the raw data, raw=FALSE, report the sample correlation matrix.

Details

Many personality and cognitive tests have a hierarchical factor structure. For demonstration purposes, it is useful to be able to create such matrices, either with population values, or sample values.

Given a matrix of item factor loadings (fload) and of loadings of these factors on a general factor (gload), we create a population correlation matrix by using the general factor law (R = F' theta F where theta = g'g).

To create sample values, we use the `mvrnorm` function from MASS.

The default is to return population correlation matrices. Sample correlation matrices are generated if n >0. Raw data are returned if raw = TRUE.

The default values for gload and fload create a data matrix discussed by Jensen and Weng, 1994.

Although written to create hierarchical structures, if the gload matrix is all 0, then a non-hierarchical structure will be generated.

Value

a matrix of correlations or a data matrix

Author(s)

William Revelle

References

http://personality-project.org/r/r.omega.html
Jensen, A.R., Weng, L.J. (1994) What is a Good g? Intelligence, 18, 231-258.

See Also

omega, schmid, ICLUST, VSS, mvrnorm

Examples

```
gload <-  gload<-matrix(c(.9,.8,.7),nrow=3)    # a higher order factor matrix
fload <-matrix(c(                    #a lower order (oblique) factor matrix
           .8,0,0,
           .7,0,.0,
           .6,0,.0,
            0,.7,.0,
            0,.6,.0,
            0,.5,0,
            0,0,.6,
            0,0,.5,
            0,0,.4),   ncol=3,byrow=TRUE)

jensen <- sim.hierarchical(gload,fload)    #the test set used by omega
round(jensen,2)

fload <- matrix(c(c(c(.9,.8,.7,.6),rep(0,20)),c(c(.9,.8,.7,.6),rep(0,20)),c(c(.9,.8,.7,.6),rep(0,20)),c(c
gload <- matrix(rep(0,5))
five.factor <- sim.hierarchical(gload,fload,500,TRUE) #create sample data set
```

---

sim.item — *Generate simulated data structures for circumplex or simple structure*

Description

Rotations of factor analysis and principal components analysis solutions typically try to represent correlation matrices as simple structured. An alternative structure, appealing to some, is a circumplex structure where the variables are uniformly spaced on the perimeter of a circle in a two dimensional space. Generating simple structure and circumplex data is straightforward, and is useful for exploring alternative solutions to affect and personality structure.

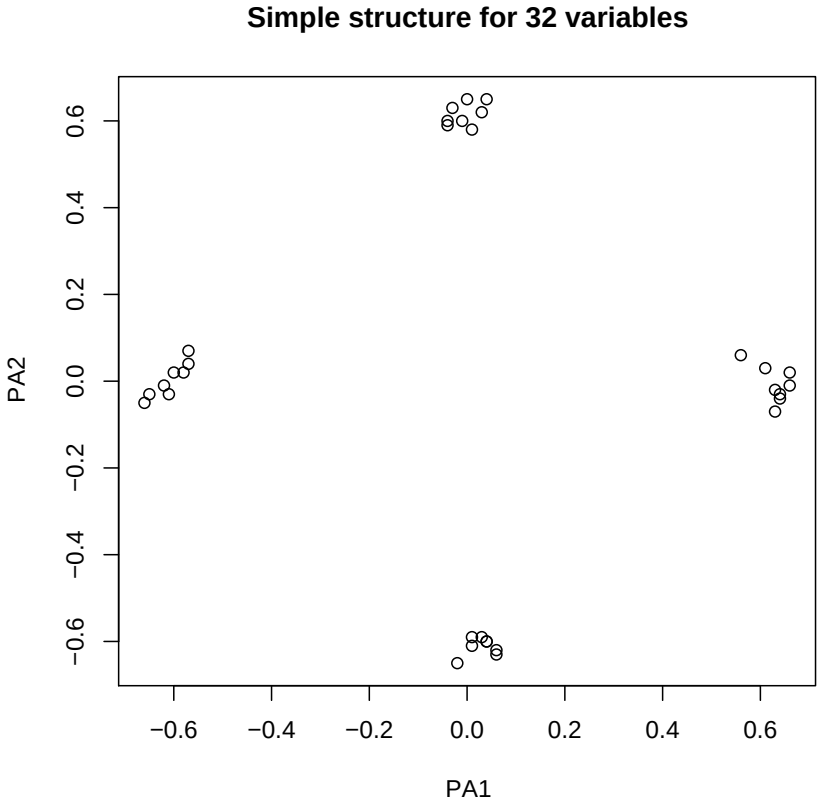

Figure 8: Simple structure is a goal for many factor rotation and extraction procedures. Data may be generated with simple or circumplex structure with varying degrees of skew and correlation. (Compare with Figure **??**)

Usage

```
sim.item(nvar = 72, nsub = 500, circum = FALSE, xloading = 0.6, yloading = 0.6, gloading = 0, x
sim.circ(nvar = 72, nsub = 500, circum = TRUE, xloading = 0.6, yloading = 0.6, gloading = 0, xb
sim.dichot(nvar = 72, nsub = 500, circum = FALSE, xloading = 0.6, yloading = 0.6, gloading = 0,
item.dichot(nvar = 72, nsub = 500, circum = FALSE, xloading = 0.6, yloading = 0.6, gloading = 0
```

Arguments

`nvar`	Number of variables to simulate
`nsub`	Number of subjects to simulate
`circum`	circum=TRUE is circumplex structure, FALSE is simple structure
`xloading`	the average loading on the first dimension
`yloading`	Average loading on the second dimension
`gloading`	Average loading on a general factor (default=0)
`xbias`	To introduce skew, how far off center is the first dimension
`ybias`	To introduce skew on the second dimension
`categorical`	continuous or categorical variables.
`low`	values less than low are forced to low (or 0 in item.dichot)
`high`	values greater than high are forced to high (or 1 in item.dichot)
`truncate`	Change all values less than cutpoint to cutpoint.
`cutpoint`	What is the cutpoint

Details

This simulation was originally developed to compare the effect of skew on the measurement of affect (see Rafaeli and Revelle, 2005). It has been extended to allow for a general simulation of affect or personality items with either a simple structure or a circumplex structure. Items can be continuous normally distributed, or broken down into n categories (e.g, -2, -1, 0, 1, 2). Items can be distorted by limiting them to these ranges, even though the items have a mean of (e.g., 1).

The addition of item.dichot allows for testing structures with dichotomous items of different difficulty (endorsement) levels. Two factor data with either simple structure or circumplex structure are generated for two sets of items, one giving a score of 1 for all items greater than the low (easy) value, one giving a 1 for all items greater than the high (hard) value. The default values for low and high are 0. That is, all items are assumed to have a 50 percent endorsement rate. To examine the effect of item difficulty, low could be -1, high 1. This will lead to item endorsements of .84 for the easy and .16 for the hard. Within each set of difficulties, the first 1/4 are assigned to the first factor factor, the second to the second factor, the third to the first factor (but with negative loadings) and the fourth to the second factor (but with negative loadings).

Value

A data matrix of (nsub) subjects by (nvar) variables.

Author(s)

William Revelle

References

Variations of a routine used in Rafaeli and Revelle, 2006; Rafaeli, E. & Revelle, W. (2006). A premature consensus: Are happiness and sadness truly opposite affects? Motivation and Emotion.

Acton, G. S. and Revelle, W. (2004) Evaluation of Ten Psychometric Criteria for Circumplex Structure. Methods of Psychological Research Online, Vol. 9, No. 1 `http://www.dgps.de/fachgruppen/methoden/mpr-online/issue22/mpr110_10.pdf`

See Also

See Also the implementation in this to generate numerous simulations. `simulation.circ`, `circ.tests` as well as other simulations (`sim.structural` `sim.hierarchical`)

Examples

```
round(cor(circ.sim(nvar=8,nsub=200)),2)
plot(factor.pa(circ.sim(16,500),2)$loadings,main="Circumplex Structure") #circumplex structure
#
#
plot(factor.pa(item.sim(16,500),2)$loadings,main="Simple Structure") #simple structure
#
cluster.plot(factor.pa(item.dichot(16,low=0,high=1),2))
```

---

sim.structural *Create correlation matrices or data matrices with a particular measurement and structural model*

Description

Structural Equation Models decompose correlation or correlation matrices into a measurement (factor) model and a structural (regression) model. sim.structural creates data sets with known measurement and structural properties. Population or sample correlation matrices with known properties are generated. Optionally raw data are produced.

It is also possible to specify a measurement model for a set of x variables separately from a set of y variables. They are then combined into one model.

Usage

```
sim.structural(fx=NULL,fy=NULL,Phi=NULL,f=NULL,n=0,raw=FALSE)
make.structural(fx=NULL,fy=NULL,Phi=NULL,f=NULL,n=0,raw=FALSE)  #deprecated
```

Arguments

`fx`	The measurement model for x
`fy`	The measurement model for y
`Phi`	The structure matrix of the latent variables
`f`	The measurement model
`n`	Number of cases to simulate. If n=0, the population matrix is returned.
`raw`	if raw=TRUE, raw data are returned as well.

Details

Given the measurement model, f and the structure model f, the model is f

Given the model, raw data are generated using the mvnorm function.

A special case of a structural model are one factor models such as parallel tests, tau equivalent tests, and congneneric tests. These may be created by letting the structure matrix = 1 and then defining a vector of factor loadings. Alternatively, make.congeneric will do the same.

Value

`model`	The implied population correlation matrix
`reliability`	The population reliability values
`r`	The sample correlation matrix
`observed`	If raw=TRUE, a sample data matrix

Author(s)

William Revelle

References

Revelle, W. (in preparation) An Introduction to Psychometric Theory with applications in R. Springer. at `http://personality-project.org/r/book/`

See Also

`make.hierarchical` for another structural model and `make.congeneric` for the one factor case.

Examples

```
fx <-matrix(c( .9,.8,.6,rep(0,4),.6,.8,-.7),ncol=2)
fy <- c(.6,.5,.4)
Phi <-matrix( c(1,0,.7,.0,1,.7,0,0,0),ncol=3)
gre.gpa <- sim.structural(fx,fy,Phi)
print(gre.gpa,2)
round(correct.cor(gre.gpa$model,gre.gpa$reliability),2)  #correct for attenuation to see structure
```

```
congeneric <- sim.structural(f=c(.9,.8,.7,.6)) # a congeneric model
congeneric
```

`sim` *Functions to simulate psychological/psychometric data*

Description

A number of functions in the psych package will generate simulated data. These functions include `sim.circ`, `sim.congeneric`, `sim.dichot`, `sim.hierarchical`, `sim.item`, `sim.structural`, and `sim.VSS`. These functions are separately documented and are listed here for ease of the help function. See each function for more detailed help.

Usage

```
sim()
```

Details

Simulation of data structures is a very useful tool in psychometric research and teaching. By knowing "truth" it is possible to see how well various algorithms can capture it. Various simulation functions in psych are:

`sim.structural` A function to combine a measurement and structural model into one data matrix. Useful for understanding structural equation models.

`sim.congeneric` A function to create congeneric items/tests for demonstrating classical test theory.

`sim.hierarchical` A function to create data with a hierarchical (bifactor) structure.

`sim.item` A function to create items that either have a simple structure or a circumplex structure.

`sim.circ` Create data with a circumplex structure.

`sim.dichot` Create dichotomous item data with a simple or circumplex structure.

Author(s)

William Revelle

References

Revelle, W. (in preparation) An Introduction to Psychometric Theory with applications in R. Springer. at `http://personality-project.org/r/book/`

See Also

See above

Examples

```
sim.congeneric()
R <- sim.hierarchical()
R
fx <- matrix(c(.9,.8,.7,rep(0,6),c(.8,.7,.6)),ncol=2)
fy <- c(.6,.5,.4)
Phi <- matrix(c(1,0,.5,0,1,.4,0,0,0),ncol=3)
print(sim.structural(fx,fy,Phi),digits=2)
```

---

sim.VSS *create VSS like data*

Description

Simulation is one of most useful techniques in statistics and psychometrics. Here we simulate a correlation matrix with a simple structure composed of a specified number of factors. Each item is assumed to have complexity one. See `circ.sim` and `item.sim` for alternative simulations.

Usage

```
sim.VSS(ncases=1000, nvariables=16, nfactors=4, meanloading=.5,dichot=FALSE,cut=0)
```

Arguments

`ncases`	number of simulated subjects
`nvariables`	Number of variables
`nfactors`	Number of factors to generate
`meanloading`	with a mean loading
`dichot`	dichot=FALSE give continuous variables, dichot=TRUE gives dichotomous variables
`cut`	if dichotomous = TRUE, then items with values > cut are assigned 1, otherwise 0.

Value

a ncases x nvariables matrix

Author(s)

William Revelle

See Also

`VSS`, `ICLUST`

Examples

```
## Not run:
simulated <- sim.VSS(1000,20,4,.6)
vss <- VSS(simulated,rotate="varimax")
VSS.plot(vss)
## End(Not run)
```

`simulation.circ` *Simulations of circumplex and simple structure*

Description

Rotations of factor analysis and principal components analysis solutions typically try to represent correlation matrices as simple structured. An alternative structure, appealing to some, is a circumplex structure where the variables are uniformly spaced on the perimeter of a circle in a two dimensional space. Generating these data is straightforward, and is useful for exploring alternative solutions to affect and personality structure.

Usage

```
simulation.circ(samplesize=c(100,200,400,800), numberofvariables=c(16,32,48,72))
```

Arguments

`samplesize`	a vector of sample sizes to simulate
`numberofvariables`	vector of the number of variables to simulate

Details

"A common model for representing psychological data is simple structure (Thurstone, 1947). According to one common interpretation, data are simple structured when items or scales have non-zero factor loadings on one and only one factor (Revelle & Rocklin, 1979). Despite the commonplace application of simple structure, some psychological models are defined by a lack of simple structure. Circumplexes (Guttman, 1954) are one kind of model in which simple structure is lacking.

"A number of elementary requirements can be teased out of the idea of circumplex structure. First, circumplex structure implies minimally that variables are interrelated; random noise does not a circumplex make. Second, circumplex structure implies that the domain in question is optimally represented by two and only two dimensions. Third, circumplex structure implies that variables do not group or clump along the two axes, as in simple structure, but rather that there are always interstitial variables between any orthogonal pair of axes (Saucier, 1992). In the ideal case, this quality will be reflected in equal spacing of variables along the circumference of the circle (Gurtman, 1994; Wiggins, Steiger, & Gaelick, 1981). Fourth, circumplex structure implies that variables have a constant radius from the center of the circle, which implies that all variables have equal communality on

the two circumplex dimensions (Fisher, 1997; Gurtman, 1994). Fifth, circumplex structure implies that all rotations are equally good representations of the domain (Conte & Plutchik, 1981; Larsen & Diener, 1992)." (Acton and Revelle, 2004)

Acton and Revelle reviewed the effectiveness of 10 tests of circumplex structure and found that four did a particularly good job of discriminating circumplex structure from simple structure, or circumplexes from ellipsoidal structures. Unfortunately, their work was done in Pascal and is not easily available. Here we release R code to do the four most useful tests:

The Gap test of equal spacing

Fisher's test of equality of axes

A test of indifference to Rotation

A test of equal Variance of squared factor loadings across arbitrary rotations.

Included in this set of functions are simple procedure to generate circumplex structured or simple structured data, the four test statistics, and a simple simulation showing the effectiveness of the four procedures.

`circ.sim.plot` compares the four tests for circumplex, ellipsoid and simple structure data as function of the number of variables and the sample size. What one can see from this plot is that although no one test is sufficient to discriminate these alternative structures, the set of four tests does a very good job of doing so. When testing a particular data set for structure, comparing the results of all four tests to the simulated data will give a good indication of the structural properties of the data.

Value

A data.frame with simulation results for circumplex, ellipsoid, and simple structure data sets for each of the four tests.

Note

The simulations default values are for sample sizes of 100, 200, 400, and 800 cases, with 16, 32, 48 and 72 items.

Author(s)

William Revelle

References

Acton, G. S. and Revelle, W. (2004) Evaluation of Ten Psychometric Criteria for Circumplex Structure. Methods of Psychological Research Online, Vol. 9, No. 1 `http://www.dgps.de/fachgruppen/methoden/mpr-online/issue22/mpr110_10.pdf`

See Also

See also `circ.tests`, `sim.circ`, `sim.structural`, `sim.hierarchical`

Examples

```
demo <- simulation.circ()
 boxplot(demo[3:14])
 title("4 tests of Circumplex Structure",sub="Circumplex, Ellipsoid, Simple Structure")
circ.sim.plot(demo[3:14])  #compare these results to real data
```

`skew` *Calculate skew or kurtosis for a vector, matrix, or data.frame*

Description

Find the skew and kurtosis for each variable in a data.frame or matrix. Unlike skew and kurtosis in e1071, this calculates a different skew for each variable or column of a data.frame/matrix.

Usage

```
skew(x, na.rm = TRUE)
kurtosi(x, na.rm = TRUE)
```

Arguments

`x`	A data.frame or matrix
`na.rm`	how to treat missing data

Details

given a matrix or data.frame x, find the skew or kurtosis for each column.

Value

if input is a matrix or data.frame, skew (kurtosi) is a vector of skews (kurtosi)

Note

The mean function supplies means for the columns of a data.frame, but the overall mean for a matrix. Mean will throw a warning for non-numeric data, but colMeans stops with non-numeric data. Thus, the function uses either mean (for data frames) or colMeans (for matrices). This is true for skew and kurtosi as well.

Author(s)

William Revelle

See Also

`describe`, `describe.by`,

Examples

```
round(skew(attitude),2)
round(kurtosi(attitude),2)
```

`smc` *Find the Squared Multiple Correlation (SMC) of each variable with the remaining variables in a matrix*

Description

The squared multiple correlation of a variable with the remaining variables in a matrix is sometimes used as initial estimates of the communality of a variable.

SMCs are also used when estimating reliability using Guttman's lambda 6 `guttman` coefficient.

The SMC is just 1 - 1/diag(R.inv) where R.inv is the inverse of R.

Usage

```
smc(R)
```

Arguments

`R`	A correlation matrix or a dataframe. In the latter case, correlations are found.

Value

a vector of squared multiple correlations.

If the matrix is not invertible, then a vector of 1s is returned

Author(s)

William Revelle

See Also

`mat.regress`, `factor.pa`

Examples

```
R <- make.hierarchical()
round(smc(R),2)
```

`structure.graph` *Draw a structural equation model specified by two measurement models and a structural model*

Description

Graphic presentations of structural equation models are a very useful way to conceptualize sem and confirmatory factor models. Given a measurement model on x (xmodel) and on y (ymodel) as well as a path model connecting x and y (phi), draw the graph. If ymodel is not specified, just draw the measurement model (xmodel + phi).

Perhaps even more usefully, the function returns a model appropriate for running directly in the *sem package* written by John Fox.

Input can be specified as matrices or the output from factanal, factor.pa, or a rotation package such as *GPArotation*.

For symbolic graphs, the input matrices can be character strings or mixtures of character strings and numeric vectors.

As an option, for those without Rgraphviz installed, structure.sem will just create the sem model and skip the graph.

Usage

```
structure.graph(xmodel,  Phi = NULL,ymodel = NULL, out.file = NULL, labels = NULL, cut = 0.3, e
structure.sem(xmodel,  Phi = NULL, ymodel = NULL,out.file = NULL, labels = NULL, cut = 0.3, err
```

Arguments

`xmodel`	a factor model on the x variables.
`ymodel`	a factor model on the y variables (can be empty)
`Phi`	A matrix of directed relationships. Lower diagonal values are drawn. If the upper diagonal values match the lower diagonal, two headed arrows are drawn. For a single, directed path, just the value may be specified.
`out.file`	name a file to send dot language instructions.
`labels`	variable labels if not specified as colnames for the matrices
`cut`	Draw paths for values > cut
`errors`	draw an error term for observerd variables
`simple`	Just draw one path per x or y variable
`regression`	Draw a regression diagram (observed variables cause Y)
`size`	page size of graphic
`node.font`	font type for graph
`edge.font`	font type for graph
`rank.direction`	Which direction should the graph be oriented

digits	Number of digits to draw
title	Title of graphic
...	other options to pass to Rgraphviz

Details

Boths function return a matrix of commands suitable for using in the sem package.

The structure.graph output can be directed to an output file for post processing using the dot graphic language.

Value

sem	a model matrix (partially) ready for input to John Fox's sem package
dotfile	If out.file is specified, a dot language file suitable for using in a dot graphics program such as graphviz or Omnigraffle.

A graphic structural diagram in the graphics window

Author(s)

William Revelle

See Also

fa.graph, omega.graph, sim.structural

Examples

```
fx <- matrix(c(.9,.8,.6,rep(0,4),.6,.8,-.7),ncol=2)
fy <- matrix(c(.6,.5,.4),ncol=1)
Phi <- matrix(c(1,0,0,0,1,0,.7,.7,1),ncol=3,byrow=TRUE)
if(require(Rgraphviz)) { f1 <- structure.graph(fx,Phi,fy)  } else {f1 <- structure.sem(fx,Phi,fy)}

#symbolic input
X2 <- matrix(c("a",0,0,"b","e1",0,0,"e2"),ncol=4)
colnames(X2) <- c("X1","X2","E1","E2")
phi2 <- diag(1,4,4)
phi2[2,1] <- phi2[1,2] <- "r"
if(require(Rgraphviz)) { f2 <- structure.graph(X2,Phi=phi2,errors=FALSE)  } else {f2 <- structure.sem(X2,

#symbolic input with error
X2 <- matrix(c("a",0,0,"b"),ncol=2)
colnames(X2) <- c("X1","X2")
phi2 <- diag(1,2,2)
phi2[2,1] <- phi2[1,2] <- "r"
if(require(Rgraphviz)) { f3 <- structure.graph(X2,Phi=phi2)  } else {f3 <- structure.sem(X2,Phi=phi2)}

#and yet another one
X6 <- matrix(c("a","b","c",rep(0,6),"d","e","f"),nrow=6)
colnames(X6) <- c("L1","L2")
rownames(X6) <- c("x1","x2","x3","x4","x5","x6")
```

```
Y3 <- matrix(c("u","w","z"),ncol=1)
colnames(Y3) <- "Y"
rownames(Y3) <- c("y1","y2","y3")
phi21 <- matrix(c(1,0,"r1",0,1,"r2",0,0,1),ncol=3)
colnames(phi21) <- rownames(phi21) <-  c("L1","L2","Y")
if(require(Rgraphviz)) {f4 <- structure.graph(X6,phi21,Y3,title="Structural model")} else {f4 <- structur

# and finally, a regression model
X7 <- matrix(c("a","b","c","d","e","f"),nrow=6)
if(require(Rgraphviz)) {f5 <- structure.graph(X7,regression=TRUE)}

#and a really messy regession model
x8 <- c("b1","b2","b3")
r8 <- matrix(c(1,"r12","r13","r12",1,"r23","r13","r23",1),ncol=3)
if(require(Rgraphviz)) {f6<- structure.graph(x8,Phi=r8,regression=TRUE)}
```

`structure.list` *Create factor model matrices from an input list*

Description

When creating a structural diagram or a structural model, it is convenient to not have to specify all of the zero loadings in a structural matrix. structure.list converts list input into a design matrix. phi.list does the same for a correlation matrix.

Usage

```
structure.list(nvars, f.list,f=NULL, f.labels = NULL, item.labels = NULL)
phi.list(nf,f.list, f.labels = NULL)
```

Arguments

`nvars`	Number of variables in the design matrix
`f.list`	A list of items included in each factor (for structure.list, or the factors that correlate with the specified factor for phi.list
`f`	prefix for parameters – needed in case of creating an X set and a Y set
`f.labels`	Names for the factors
`item.labels`	Item labels
`nf`	Number of factors in the phi matrix

Details

This is almost self explanatory. See the examples.

Value

`factor.matrix`	a matrix of factor loadings to model

See Also

`structure.graph`

Examples

```
fx <- structure.list(9,list(F1=c(1,2,3),F2=c(4,5,6),F3=c(7,8,9)))
fy <- structure.list(3,list(Y=c(1,2,3)),"Y")
phi <- phi.list(4,list(F1=c(4),F2=c(1,4),F3=c(2),F4=c(1,2,3)))
fx
phi
fy
```

`super.matrix` *Form a super matrix from two sub matrices.*

Description

Given the matrices nXm, and jYk, form the super matrix of dimensions (n+j) and (m+k) with with elements x and y along the super diagonal. Useful when considering structural equations. The measurement models x and y can be combined into a larger measurement model of all of the variables.

Usage

```
super.matrix(x, y)
```

Arguments

x	A n x m matrix
y	A j x k matrix

Value

A (n+j) x (m +k) matrix with appropriate row and column names

Author(s)

William Revelle

See Also

`sim.structural`,`structure.graph`

Examples

```
mx <- matrix(c(.9,.8,.7,rep(0,4),.8,.7,.6),ncol=2)
my <- matrix(c(.6,.5,.4))
colnames(mx) <- paste("X",1:dim(mx)[2],sep="")
rownames(mx) <- paste("Xv",1:dim(mx)[1],sep="")
colnames(my) <- "Y"
rownames(my) <- paste("Yv",1:3,sep="")
super.matrix(mx,my)
```

`table2matrix` *Convert a table with counts to a matrix or data.frame representing those counts.*

Description

Some historical sets are reported as summary tables of counts in a limited number of bins. Transforming these tables to data.frames representing the original values is useful for pedagogical purposes. (E.g., transforming the original Galton table of height x cubits in order to demonstrate regression.) The column and row names must be able to be converted to numeric values.

Usage

```
table2matrix(x, labs = NULL)
table2df(x, labs = NULL)
```

Arguments

`x`	A two dimensional table of counts with row and column names that can be converted to numeric values.
`labs`	Labels for the rows and columns. These will be used for the names of the two columns of the resulting matrix

Details

The original Galton (1888) of heights by cubits (arm length) is in tabular form. To show this as a correlation or as a scatter plot, it is useful to convert the table to a matrix or data frame of two columns.

Value

A matrix (or data.frame) of sum(x) rows and two columns.

Author(s)

William Revelle

See Also

cubits

Examples

```
data(cubits)
cubit <- table2matrix(cubits,labs=c("height","cubit"))
describe(cubit)
ellipses(cubit,n=1)
```

---

test.psych *Testing of functions in the psych package*

Description

Test to make sure the psych functions run on basic test data sets

Usage

```
test.psych(first=1,last=5,short=TRUE)
```

Arguments

first	first=1: start with dataset first
last	last=5: test for datasets until last
short	short=TRUE - don't return any analyses

Details

When modifying the psych package, it is useful to make sure that adding some code does not break something else. The test.psych function tests the major functions on various standard data sets. It also shows off a number of the capabilities of the psych package.

Uses 5 standard data sets:
USArrests Violent Crime Rates by US State (4 variables)
attitude The Chatterjee-Price Attitude Data
Harman23.cor$cov Harman Example 2.3 8 physical measurements
Harman74.cor$cov Harman Example 7.4 24 mental measurements
ability.cov$cov 8 Ability and Intelligence Tests

Value

out	if short=FALSE, then list of the output from all functions tested

Warning

Warning messages will be thrown by fa.parallel and sometimes by factor.pa for random datasets.

Note

Although test.psych may be used as a quick demo of the various functions in the psych packge, in general, it is better to try the specific functions themselves. The main purpose of test.psych is to make sure functions throw error messages or correct for weird conditions.

The datasets tested are part of the standard R data sets and represent some of the basic problems encountered.

Author(s)

William Revelle

Examples

```
test <- test.psych()
```

---

thurstone *Thurstone Case V scaling*

Description

Thurstone Case V scaling allows for a scaling of objects compared to other objects. As one of the cases considered by Thurstone, Case V makes the assumption of equal variances and uncorrelated distributions.

Usage

```
thurstone(x, ranks = FALSE, digits = 2)
```

Arguments

x	A square matrix or data frame of preferences, or a rectangular data frame or matrix rank order choices.
ranks	TRUE if rank orders are presented
digits	number of digits in the goodness of fit

Details

Louis L. Thurstone was a pioneer in psychometric theory and measurement of attitudes, interests, and abilities. Among his many contributions was a systematic analysis of the process of comparative judgment (thurstone, 1927). He considered the case of asking subjects to successively compare pairs of objects. If the same subject does this repeatedly, or if subjects act as random replicates of each other, their judgments can be thought of as sampled from a normal distribution of underlying (latent) scale scores for each object, Thurstone proposed that the comparison between the value of two objects could be represented as representing the differences of the average value for each object compared to the standard deviation of the differences between objects. The basic model is that each

item has a normal distribution of response strength and that choice represents the stronger of the two response strengths. A justification for the normality assumption is that each decision represents the sum of many independent inputs and thus, through the central limit theorem, is normally distributed.

Thurstone considered five different sets of assumptions about the equality and independence of the variances for each item (Thurston, 1927). Torgerson expanded this analysis slightly by considering three classes of data collection (with individuals, between individuals and mixes of within and between) crossed with three sets of assumptions (equal covariance of decision process, equal correlations and small differences in variance, equal variances).

The data may be either a square matrix of dataframe of preferences (as proportions with the probability of the column variable being chosen over the row variable) or a matrix or dataframe of rank orders (1 being prefered to 2, etc.)

Value

`GF`	Goodness of fit 1 = 1 - sum(squared residuals/squared original) for lower off diagonal. Goodness of fit 2 = 1 - sum(squared residuals/squared original) for full matrix.
`residual`	square matrix of residuals (of class dist)
`data`	The original choice data
...	

Author(s)

William Revelle

References

Thurstone, L. L. (1927) A law of comparative judgments. Psychological Review, 34, 273-286.

Revelle, W. An introduction to psychometric theory with applications in R. (in preparation), Springer. `http://personality-project.org/r/book`

Examples

```
data(vegetables)
thurstone(veg)
```

tr *Find the trace of a square matrix*

Description

Hardly worth coding, if it didn't appear in so many formulae in psychometrics, the trace of a (square) matrix is just the sum of the diagonal elements.

Usage

```
tr(m)
```

Arguments

m	A square matrix

Details

The tr function is used in various matrix operations and is the sum of the diagonal elements of a matrix.

Value

The sum of the diagonal elements of a square matrix.
i.e. tr(m) <- sum(diag(m)).

Examples

```
m <- matrix(1:16,ncol=4)
m
tr(m)
```

vegetables *Paired comparison of preferences for 9 vegetables*

Description

A classic data set for demonstrating Thurstonian scaling is the preference matrix of 9 vegetables from Guilford (1954). Used by Guiford, Nunnally, and Nunally and Bernstein, this data set allows for examples of basic scaling techniques.

Usage

```
data(vegetables)
```

Format

A data frame with 9 choices on the following 9 vegetables. The values reflect the perecentage of times where the column entry was preferred over the row entry.

`Turn` Turnips

`Cab` Cabbage

`Beet` Beets

`Asp` Asparagus

`Car` Carrots

`Spin` Spinach

`S.Beans` String Beans

`Peas` Peas

`Corn` Corn

Details

Louis L. Thurstone was a pioneer in psychometric theory and measurement of attitudes, interests, and abilities. Among his many contributions was a systematic analysis of the process of comparative judgment (thurstone, 1927). He considered the case of asking subjects to successively compare pairs of objects. If the same subject does this repeatedly, or if subjects act as random replicates of each other, their judgments can be thought of as sampled from a normal distribution of underlying (latent) scale scores for each object, Thurstone proposed that the comparison between the value of two objects could be represented as representing the differences of the average value for each object compared to the standard deviation of the differences between objects. The basic model is that each item has a normal distribution of response strength and that choice represents the stronger of the two response strengths. A justification for the normality assumption is that each decision represents the sum of many independent inputs and thus, through the central limit theorem, is normally distributed.

Thurstone considered five different sets of assumptions about the equality and independence of the variances for each item (Thurston, 1927). Torgerson expanded this analysis slightly by considering three classes of data collection (with individuals, between individuals and mixes of within and between) crossed with three sets of assumptions (equal covariance of decision process, equal correlations and small differences in variance, equal variances).

This vegetable data set is used by Guilford and by Nunnally to demonstrate Thurstonian scaling.

Source

Guilford, J.P. (1954) Psychometric Methods. McGraw-Hill, New York.

References

Nunnally, J. C. (1967). Psychometric theory., McGraw-Hill, New York.

Revelle, W. An introduction to psychometric theory with applications in R. (in preparation), Springer. `http://personality-project.org/r/book`

See Also

thurstone

Examples

```
data(vegetables)
thurstone(veg)
```

VSS.parallel *Compare real and random VSS solutions*

Description

Another useful test for the number of factors is when the eigen values of a random matrix are greater than the eigen values of a a real matrix. Here we show VSS solutions to random data.

Usage

```
VSS.parallel(ncases, nvariables,scree=FALSE,rotate="none")
```

Arguments

ncases	Number of simulated cases
nvariables	number of simulated variables
scree	Show a scree plot for random data – see omega
rotate	rotate="none" or rotate="varimax"

Value

VSS like output to be plotted by VSS.plot

Author(s)

William Revelle

References

Very Simple Structure (VSS)

See Also

fa.parallel, VSS.plot, ICLUST, omega

Examples

```
#VSS.plot(VSS.parallel(200,24))
```

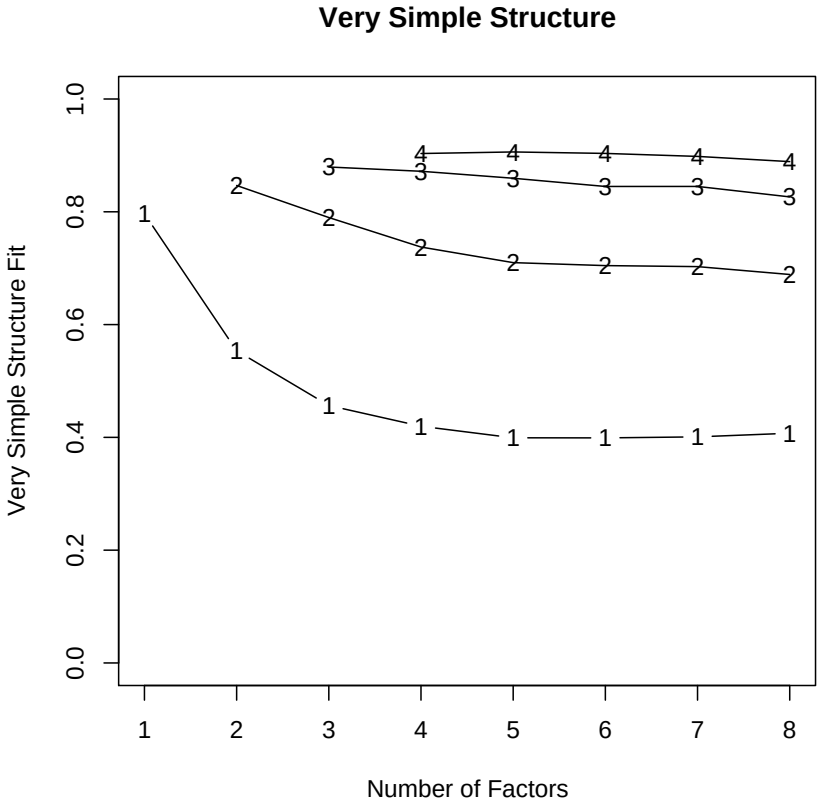

Figure 9: The Very Simple Structure criterion plots goodness of fit as a function of the number of factors extracted and the factorial complexity for each item/test. Note how a complexity one solution best fits the data if only one factor is extracted, but a complexity three solution is optimized at 4 factors.

`VSS.plot` *Plot VSS fits*

Description

The Very Simple Structure criterion (`VSS`) for estimating the optimal number of factors is plotted as a function of the increasing complexity and increasing number of factors.

Usage

```
VSS.plot(x, title = "Very Simple Structure", line = FALSE)
```

Arguments

`x`	output from VSS
`title`	any title
`line`	connect different complexities

Details

Item-factor models differ in their "complexity". Complexity 1 means that all except the greatest (absolute) loading for an item are ignored. Basically a cluster model (e.g., `ICLUST`). Complexity 2 implies all except the greatest two, etc.

Different complexities can suggest different number of optimal number of factors to extract. For personality items, complexity 1 and 2 are probably the most meaningful.

The Very Simple Structure criterion will tend to peak at the number of factors that are most interpretable for a given level of complexity. Note that some problems, the most interpretable number of factors will differ as a function of complexity. For instance, when doing the Harman 24 psychological variable problems, an unrotated solution of complexity one suggests one factor (g), while a complexity two solution suggests that a four factor solution is most appropriate. This latter probably reflects a bi-factor structure.

For examples of VSS.plot output, see `http://personality-project.org/r/r.vss.html`

Value

A plot window showing the VSS criterion varying as the number of factors and the complexity of the items.

Author(s)

Maintainer: William Revelle ⟨revelle@northwestern.edu⟩

References

`http://personality-project.org/r/r.vss.html`

See Also

VSS, ICLUST, omega

Examples

```
test.data <- Harman74.cor$cov
my.vss <- VSS(test.data)         #suggests that 4 factor complexity two solution is optimal
VSS.plot(my.vss,title="VSS of Holzinger-Harmon problem")                #see the graphics window
```

VSS.scree *Plot a scree test*

Description

Cattell's scree test is one of most simple ways of testing the number of components in a correlation matrix. Here we plot the eigen values of a correlation matrix.

Usage

```
VSS.scree(rx, main = "scree plot")
```

Arguments

rx	a correlation matrix or a data matrix. If data, then correlations are found using pairwise deletions.
main	Title

Author(s)

William Revelle

References

http://personality-project.org/r/vss.html

See Also

VSS.plot, ICLUST, omega

Examples

```
#VSS.scree(attitude)
#VSS.scree(cor(attitude)
```

`VSS` *Apply the Very Simple Structure and MAP criteria to determine the appropriate number of factors.*

Description

There are multiple ways to determine the appropriate number of factors in exploratory factor analysis. Routines for the Very Simple Structure (VSS) criterion allow one to compare solutions of varying complexity and for different number of factors. Graphic output indicates the "optimal" number of factors for different levels of complexity. The Velicer MAP criterion is another good choice.

Usage

```
VSS(x, n = 8, rotate = "varimax", diagonal = FALSE, pc = "pa", n.obs=NULL,plot=TRUE,title="Very
```

Arguments

`x`	a correlation matrix or a data matrix
`n`	Number of factors to extract – should be more than hypothesized!
`rotate`	what rotation to use c("none", "varimax", "oblimin","promax")
`diagonal`	Should we fit the diagonal as well
`pc`	pc="pa" Principal Axis Factor Analysis, pc="mle" Maximum Likelihood FA, pc="pc" Principal Components"
`n.obs`	Number of observations if doing a factor analysis of correlation matrix. This value is ignored by VSS but is necessary for the ML factor analysis package.
`plot`	plot=TRUE Automatically call VSS.plot with the VSS output, otherwise don't plot
`title`	a title to be passed on to VSS.plot
`...`	parameters to pass to the factor analysis program The most important of these is if using a correlation matrix is covmat= xx

Details

Determining the most interpretable number of factors from a factor analysis is perhaps one of the greatest challenges in factor analysis. There are many solutions to this problem, none of which is uniformly the best. "Solving the number of factors problem is easy, I do it everyday before breakfast. But knowing the right solution is harder" (Kaiser, 195x).

Techniques most commonly used include

1) Extracting factors until the chi square of the residual matrix is not significant.

2) Extracting factors until the change in chi square from factor n to factor n+1 is not significant.

3) Extracting factors until the eigen values of the real data are less than the corresponding eigen values of a random data set of the same size (parallel analysis) `fa.parallel`.

4) Plotting the magnitude of the successive eigen values and applying the scree test (a sudden drop in eigen values analogous to the change in slope seen when scrambling up the talus slope of a mountain and approaching the rock face.

5) Extracting principal components until the eigen value <1.

6) Extracting factors as long as they are interpetable.

7) Using the Very Structure Criterion (VSS).

8) Using Wayne Velicer's Minimum Average Partial (MAP) criterion.

Each of the procedures has its advantages and disadvantages. Using either the chi square test or the change in square test is, of course, sensitive to the number of subjects and leads to the nonsensical condition that if one wants to find many factors, one simlpy runs more subjects. Parallel analysis is partially sensitive to sample size in that for large samples the eigen values of random factors will be very small. The scree test is quite appealling but can lead to differences of interpretation as to when the scree "breaks". The eigen value of 1 rule, although the default for many programs, seems to be a rough way of dividing the number of variables by 3. Extracting interpretable factors means that the number of factors reflects the investigators creativity more than the data. VSS, while very simple to understand, will not work very well if the data are very factorially complex. (Simulations suggests it will work fine if the complexities of some of the items are no more than 2).

Most users of factor analysis tend to interpret factor output by focusing their attention on the largest loadings for every variable and ignoring the smaller ones. Very Simple Structure operationalizes this tendency by comparing the original correlation matrix to that reproduced by a simplified version (S) of the original factor matrix (F). R = SS' + U2. S is composed of just the c greatest (in absolute value) loadings for each variable. C (or complexity) is a parameter of the model and may vary from 1 to the number of factors.

The VSS criterion compares the fit of the simplified model to the original correlations: VSS = 1 -sumsquares(r*)/sumsquares(r) where R* is the residual matrix R* = R - SS' and r* and r are the elements of R* and R respectively.

VSS for a given complexity will tend to peak at the optimal (most interpretable) number of factors (Revelle and Rocklin, 1979).

Although originally written in Fortran for main frame computers, VSS has been adapted to micro computers (e.g., Macintosh OS 6-9) using Pascal. We now release R code for calculating VSS.

Note that if using a correlation matrix (e.g., my.matrix) and doing a factor analysis, the parameters n.obs should be specified for the factor analysis: e.g., the call is VSS(my.matrix,n.obs=500). Otherwise it defaults to 1000.

Wayne Velicer's MAP criterion has been added as an additional test for the optimal number of components to extract. Note that VSS and MAP will not always agree as to the optimal number.

A variety of rotation options are available. These include varimax, promax, and oblimin. Others can be added. Suggestions are welcome.

Value

A data.frame with entries: map: Velicer's MAP values (lower values are better)
dof: degrees of freedom (if using FA)
chisq: chi square (from the factor analysis output (if using FA)
prob: probability of residual matrix > 0 (if using FA)
sqresid: squared residual correlations
fit: factor fit of the complete model
cfit.1: VSS fit of complexity 1
cfit.2: VSS fit of complexity 2
...
cfit.8: VSS fit of complexity 8
cresidiual.1: sum squared residual correlations for complexity 1
...: sum squared residual correlations for complexity 2 ..8

Author(s)

William Revelle

References

`http://personality-project.org/r/vss.html`, Revelle, W. An introduction to psychometric theory with applications in R (in prep) Springer. Draft chapters available at `http://personality-project.org/r/book/`

Revelle, W. and Rocklin, T. 1979, Very Simple Structure: an Alternative Procedure for Estimating the Optimal Number of Interpretable Factors, Multivariate Behavioral Research, 14, 403-414. `http://personality-project.org/revelle/publications/vss.pdf`

Velicer, W. (1976) Determining the number of components from the matrix of partial correlations. Psychometrika, 41, 321-327.

See Also

`VSS.plot`, `ICLUST`, `omega`, `fa.parallel`

Examples

```
test.data <- Harman74.cor$cov
my.vss <- VSS(test.data,title="VSS of 24 mental tests")
#print(my.vss[,1:12],digits =2)
#VSS.plot(my.vss, title="VSS of 24 mental tests")

#now, some simulated data with two factors
VSS(circ.sim(nvar=24),pc="mle" ,title="VSS of 24 circumplex variables")
VSS(item.sim(nvar=24),pc="mle" ,title="VSS of 24 simple structure variables")
```

`winsor` *Find the Winsorized scores or means for a vector, matrix, or data.frame*

Description

Among the robust estimates of central tendency are trimmed means and Winsorized means. This function finds the Winsorized mean. The top and bottom trim values are given values of the trimmed and 1- trimmed quantiles. Then means are found.

Usage

```
winsor(x, trim = 0.2, na.rm = TRUE)
winsor.means(x, trim = 0.2, na.rm = TRUE)
```

Arguments

`x`	A data vector, matrix or data frame
`trim`	Percentage of data to move from the top and bottom of the distributions
`na.rm`	Missing data are removed

Details

Among the many robust estimates of central tendency, some recommend the Winsorized mean. Rather than just dropping the top and bottom trim percent, these extreme values are replaced with values at the trim and 1- trim quantiles.

Value

A scalar or vector of winsorized scores or winsorized means (depending upon the call).

Author(s)

William Revelle with modifications suggested by Joe Paxton and a further correction added (January, 2009) to preserve the original order for the winsor case.

References

Wilcox, Rand R. (2005) Introduction to robust estimation and hypothesis testing. Elsevier/Academic Press. Amsterdam ; Boston.

See Also

`interp.median`

Examples

```
data(sat.act)
winsor.means(sat.act) #compare with the means of the winsorized scores
y <- winsor(sat.act)
describe(y)
xy <- data.frame(sat.act,y)
pairs.panels(xy) #to see the effect of winsorizing
x <- matrix(1:100,ncol=5)
winsor(x)
winsor.means(x)
y <- 1:11
winsor(y,trim=.5)
```

`Yule` *From a two by two table, find the Yule coefficients of association, convert to phi, or polychoric, recreate table the table to create the Yule coefficient.*

Description

One of the many measures of association is the Yule coefficient. Given a two x two table of counts

a	b
c	d

Yule Q is (ad - bc)/(ad+bc).
Conceptually, this is the number of pairs in agreement (ad) - the number in disagreement (bc) over the total number of paired observations.
ad/bc is the odds ratio and Q = (OR-1)/(OR+1)
Yule's coefficient of colligation is Y = (sqrt(OR) - 1)/(sqrt(OR)+1) Yule.inv finds the cell entries for a particular Q and the marginals (a+b,c+d,a+c, b+d). This is useful for converting old tables of correlations into more conventional `phi` or polychoric correlations. Yule2phi and Yule2poly convert the Yule Q with set marginals to the correponding phi or polychoric correlation.

Usage

```
Yule(x,Y=FALSE)
Yule.inv(Q,m)
Yule2phi(Q,m)
Yule2poly(Q,m)
```

Arguments

`x`	A vector of four elements or a two by two matrix

`Y`	Y=TRUE return Yule's Y coefficient of colligation
`Q`	The Yule coefficient
`m`	A two x two matrix of marginals or a four element vector of marginals

Details

Yule developed two measures of association for two by two tables. Both are functions of the odds ratio

Value

`Q`	The Yule Q coefficient
`R`	A two by two matrix of counts

Note

Currently done by using the optimize function, but presumably could be redone by solving a quadratic equation.

Author(s)

William Revelle

References

Yule, G. Uday (1912) On the methods of measuring association between two attributes. Journal of the Royal Statistical Society, LXXV, 579-652

See Also

See Also as `phi`, `Yule2poly.matrix`, `Yule2phi.matrix`

Examples

```
Nach <- matrix(c(40,10,20,50),ncol=2,byrow=TRUE)
Yule(Nach)
Yule.inv(.81818,c(50,70,60,60))
Yule2phi(.81818,c(50,70,60,60))
if(require(polycor)) Yule2poly(.81818,c(50,70,60,60))
phi(Nach)  #much less
```

Index