Package 'psych'

June 24, 2011

Version 1.0.99

Date 2011-06-15

Title Procedures for Psychological, Psychometric, and Personality Research

Author William Revelle <revelle@northwestern.edu>

Maintainer William Revelle <revelle@northwestern.edu>

Description A number of routines for personality, psychometrics and experimental psychology. Functions are primarily for scale construction using factor analysis, cluster analysis and reliability analysis, although others provide basic descriptive statistics. Item Response Theory is done using factor analysis of tetrachoric and polychoric correlations. Functions for simulating particular item and test structures are included. Several functions serve as a useful front end for structural equation modeling. Graphical displays of path diagrams, factor analysis and structural equation models are created using basic graphics. Some of the functions are written to support a book on psychometrics as well as publications in personality research. For more information, see the personality-project.org/r webpage.

License GPL (>= 2)

Suggests MASS, GPArotation, graph, Rgraphviz,mvtnorm, polycor, sem,Rcsdp, lavaan

LazyData true

URL http://personality-project.org/r, http://personality-project.org/r/psych.manual.pdf

R topics documented:

`00.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. Use help(package="psych") for a list of all functions. Two vignettes are included as part of the package. The overview provides examples of using psych in many applications.

Psychometric applications include routines (`fa` for principal axes (`factor.pa`), minimum residual (minres: `factor.minres`), and weighted least squares (`link{factor.wls}` factor analysis as well as functions to do Schmid Leiman transformations (`schmid`) to transform a hierarchical factor structure into a bifactor solution. Factor or components transformations to a target matrix include the standard Promax transformation (`Promax`), a transformation to a cluster target, or to any simple target matrix (`target.rot`) as well as the ability to call many of

the GPArotation functions. Functions for determining the number of factors in a data matrix include Very Simple Structure (VSS) and Minimum Average Partial correlation (MAP). An alternative approach to factor analysis is Item Cluster Analysis (ICLUST). Reliability coefficients alpha (score.items, score.multiple.choice), beta (ICLUST) and McDonald's omega (omega and omega.graph) as well as 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. The tetrachoric, polychoric and irt.fa functions are used to find 2 parameter descriptions of item functioning.

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. These are discussed in more detail in the vignette (psych_for_sem).

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, and 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

Two vignettes (overview.pdf) and psych_for_sem.pdf) are useful introductions to the package. They may be found as vignettes in R or may be downloaded from http://personality-project.org/r/book/overview.pdf and http://personality-project.org/r/book/psych_for_sem.pdf.

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). Those files have now been replaced with 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. corr.test will find the significance values for a matrix of correlations.

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), principal axis (factor.pa) or minimum residual (factor.minres) factor analysis (all part of the fa function) 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 forming maximally consistent and independent item composites. Graphical output from ICLUST.graph uses the Graphviz dot language and allows one to write files suitable for Graphviz. If Rgraphviz is available, these graphs can be done in R.

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 (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.

This can also be examined by applying irt.fa Item Response Theory techniques using factor analysis of the tetrachoric or polychoric correlation matrices and converting the results into the standard two parameter parameterization of item difficulty and item discrimination. Information functions for the items suggest where they are most effective.

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. alpha will report several reliablity estimates as well as item-whole correlations for items forming a single scale, score.items will score data sets on multiple scales, reporting the scale scores, item-scale and scale-scale correlations, as well as coefficient alpha, alpha-1 and G6+. 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. sim.anova produces data simulating a 3 way analysis of variance (ANOVA) or linear model with or with out repeated measures. sim.item creates simple structure data, sim.circ will produce circumplex structured data, sim.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. sim.structural will produce correlation matrices and data matrices to match general structural models. (See the vignette).

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:	1.0-97
Date:	2011-05-15
License:	GPL version 2 or newer

Index:

Useful data entry and descriptive statistics

headtail	combines the head and tail functions for showing data sets
pairs.panels	SPLOM and correlations for a data matrix
corr.test	Correlations, sample sizes, and p values for a data matrix
cor.plot	graphically show the size of correlations in a correlation 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

fa	Combined function for principal axis, minimum residual, weighted least squares, and maximum li
factor.pa	Do a principal Axis factor analysis (deprecated)
factor.minres	Do a minimum residual factor analysis (deprecated)
factor.wls	Do a weighted least squares factor analysis (deprecated)
fa.graph	Show the results of a factor analysis or principal components analysis graphically
fa.diagram	Show the results of a factor analysis without using Rgraphviz
fa.sort	Sort a factor or principal components output
fa.extension	Apply the Dwyer extension for factor loadingss
principal	Do an eigen value decomposition to find the principal components of a matrix
fa.parallel	Scree test and Parallel analysis
fa.parallel.poly	Scree test and Parallel analysis for polychoric matrices
factor.scores	Estimate factor scores given a data matrix and factor loadings
guttman	8 different measures of reliability (6 from Guttman (1945)
irt.fa	Apply factor analysis to dichotomous items to get IRT parameters
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
polychoric	Find the polychoric correlations for items and find item thresholds (uses J. Fox's polycor)
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 (uses Rgraphviz)
partial.r	Partial variables from a correlation matrix
predict	Predict factor/component scores for new data
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
set.cor	Find Cohen's set correlation between two sets of variables
smc	Find the Squared Multiple Correlation (used for initial communality estimates)
tetrachoric	Find tetrachoric correlations and item thresholds
polyserial	Find polyserial and biserial correlations for item validity studies
mixed.cor	Form a correlation matrix from continuous, polytomous, and dichotomous items
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

Functions for reliability analysis (some are listed above as well).

alpha	Find coefficient alpha and Guttman Lambda 6 for a scale (see also score.items)
guttman	8 different measures of reliability (6 from Guttman (1945)
omega	Calculate the omega estimates of reliability (requires the GPArotation package)
omegaSem	Calculate the omega estimates of reliability using a Confirmatory model (requires the sem package)
ICC	Intraclass correlation coefficients
score.items	Combine items into multiple scales and find alpha
glb.algebraic	The greates lower bound found by an algebraic solution (requires Rcsdp). Written by Andreas Moeltner

Procedures particularly useful for Synthetic Aperture Personality Assessment

alpha	Find coefficient alpha and Guttman Lambda 6 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
fa	Do a minimal residual or principal axis factor analysis and estimate factor scores
fa.extension	Extend a factor analysis to a set of new variables
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
polyserial	polyserial and biserial correlations with massive missing data
tetrachoric	Find tetrachoric correlations and item thresholds

Functions for generating simulated data sets

sim	The basic simulation functions
sim.anova	Generate 3 independent variables and 1 or more dependent variables for demonstrating ANOVA and lm
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.minor	Simulate nfact major and nvar/2 minor factors
sim.structural	Generate a multifactorial structural model

sim.irt	Generate data for a 1, 2, 3 or 4 parameter logistic 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) – deprecated

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 transforma
ICLUST.graph	Draw the tree diagram from ICLUST

Graphical functions that do not require Rgraphviz

diagram	A general set of diagram functions.
structure.diagram	Draw a sem or regression graph
fa.diagram	Draw the factor structure from a factor or principal components analysis
omega.diagram	Draw the factor structure from an omega analysis (either with or without the Schmid Leiman transfor
ICLUST.diagram	Draw the tree diagram from ICLUST
plot.psych	A call to plot various types of output (e.g. from irt.fa, fa, omega, iclust

Circular statistics (for circadian data analysis)

circadian.cor	Find the correlation with e.g., mood and time of day
circadian.linear.cor	Correlate a circular value with a linear value
circadian.mean	Find the circular mean of each column of a a data set
cosinor	Find the best fitting phase angle for a circular data set

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 values
polychor.matrix	Use John Fox's polycor to create a matrix of polychoric correlations from a matrix of Yule correlations
scaling.fits	Compares alternative scaling solutions and gives goodness of fits
scrub	Basic data cleaning
tetrachor	Finds tetrachoric correlations
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
Thurstone	8 different data sets with a bifactor structure
cities	The airline distances between 11 cities (used to demonstrate MDS)
epi.bfi	13 personality scales
iqitems	14 multiple choice iq items
msq	75 mood items
sat.act	Self reported ACT and SAT Verbal and Quantitative scores by age and gender
Tucker	Correlation matrix from Tucker
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. Although the o

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 down-

loading 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 but have alternatives using the diagram functions. 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
glb.algebraic	Rcsdp

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()
```

`affect` *Two data sets of affect and arousal scores as a function of personality and movie conditions*

Description

A recurring question in the study of affect is the proper dimensionality and the relationship to various personality dimensions. Here are two data sets taken from studies of mood and arousal using movies to induce affective states.

Usage

```
data(affect)
```

Details

These are data from two studies conducted in the Personality, Motivation and Cognition Laboratory at Northwestern University. Both studies used a similar methodology:

Collection of pretest data using 5 scales from the Eysenck Personality Inventory and items taken from the Motivational State Questionnaire. In addition, state and trait anxiety measures were given. In the “maps" study, the Beck Depression Inventory was given also.

Then subjects were randomly assigned to one of four movie conditions: Frontline. A documentary about the liberation of the Bergen-Belsen concentration camp. Halloween. A horror film. 3: National Geographic, a nature film about the Serengeti plain. 4: Parenthood. A comedy. Each film clip was shown for 9 minutes. Following this the MSQ was given again.

Data from the MSQ was scored for Energetic and Tense Arousal (EA and TA) as well as Positive and Negative Affect (PA and NA).

Study flat had 170 participants, study maps had 160.

These studies are described in more detail in various publications from the PMC lab. In particular, Revelle and Anderson, 1997 and Rafaeli and Revelle (2006).

Source

Data collecte at the Personality, Motivation, and Cognition Laboratory, Northwestern University.

References

William Revelle and Kristen Joan Anderson (1997) Personality, motivation and cognitive performance: Final report to the Army Research Institute on contract MDA 903-93-K-0008

Rafaeli, Eshkol and Revelle, William (2006), A premature consensus: Are happiness and sadness truly opposite affects? Motivation and Emotion, 30, 1, 1-12.

Examples

```
data(affect)
describe(flat)
pairs.panels(flat[15:18],bg=c("red","black","white","blue")[maps$Film],pch=21,main="Affec
describe(maps)
pairs.panels(maps[14:17],bg=c("red","black","white","blue")[maps$Film],pch=21,main="Affec
```

---

alpha *Find two estimates of reliability: Cronbach's alpha and Guttman's Lambda 6.*

Description

Internal consistency measures of reliability range from ω_h to α to ω_t. This function reports two estimates: Cronbach's coefficient α and Guttman's λ_6. Also reported are item - whole correlations, α if an item is omitted, and item means and standard deviations.

Usage

```
alpha(x, keys=NULL,cumulative=FALSE, title=NULL, max=10,na.rm = TRUE,check.keys=
```

Arguments

x	A data.frame or matrix of data, or a covariance or correlation matrix
keys	If some items are to be reversed keyed, then the direction of all items must be specified in a keys vector
title	Any text string to identify this run
cumulative	should means reflect the sum of items or the mean of the items. The default value is means.
max	the number of categories/item to consider if reporting category frequencies. Defaults to 10, passed to link{response.frequencies}
na.rm	The default is to remove missing values and find pairwise correlations
check.keys	if TRUE, then find the first principal component and reverse key items with negative loadings. Give a warning if this happens.

Details

Alpha is one of several estimates of the internal consistency reliability of a test.

Surprisingly, 105 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.

α (Cronbach, 1951) is the same as Guttman's $\lambda 3$ (Guttman, 1945) and may be found by

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

Perhaps because it is so easy to calculate and is available in most commercial programs, alpha 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 β (Revelle, 1979; see ICLUST) and ω_h (see omega) are more appropriate estimates of the general factor saturation. ω_t (see omega) is a better estimate of the reliability of the total test.

Guttman's Lambda 6 (G6) 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

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

The squared multiple correlation is a lower bound for the item communality and as the number of items increases, becomes a better estimate.

G6 is also sensitive to lumpyness in the test and should not be taken as a measure of unifactorial structure. For lumpy tests, it will be greater than alpha. For tests with equal item loadings, alpha > G6, but if the loadings are unequal or if there is a general factor, G6 > alpha. alpha is a generalization of an earlier estimate of reliability for tests with dichotomous items developed by Kuder and Richardson, known as KR20, and a shortcut approximation, KR21. (See Revelle, in prep).

Alpha and G6 are both positive functions 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, raw alpha is sensitive to differences in the item variances. Standardized alpha is based upon the correlations rather than the covariances.

More complete reliability analyses of a single scale can be done using the `omega` function which finds ω_h and ω_t based upon a hierarchical factor analysis.

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.

Three alternative item-whole correlations are reported, two are conventional, one unique. r is the correlation of the item with the entire scale, not correcting for item overlap. r.drop is the correlation of the item with the scale composed of the remaining items. Although both of these are conventional statistics, they have the disadvantage that a) item overlap inflates the first and b) the scale is different for each item when an item is dropped. Thus, the third alternative, r.cor, corrects for the item overlap by subtracting the item variance but then replaces this with the best estimate of common variance, the smc.

Value

`total`	a list containing
`raw_alpha`	alpha based upon the covariances
`std.alpha`	The standarized alpha based upon the correlations
`G6(smc)`	Guttman's Lambda 6 reliability
`average_r`	The average interitem correlation
`mean`	For data matrices, the mean of the scale formed by summing the items
`sd`	For data matrices, the standard deviation of the total score
`alpha.drop`	A data frame with all of the above for the case of each item being removed one by one.
`item.stats`	A data frame including
`r`	The correlation of each item with the total score (not corrected for item overlap)
`r.cor`	Item whole correlation corrected for item overlap and scale reliability
`r.drop`	Item whole correlation for this item against the scale without this item
`mean`	for data matrices, the mean of each item
`sd`	For data matrices, the standard deviation of each item
`response.freq`	For data matrices, the frequency of each item response (if less than 20)

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. (in preparation) An introduction to psychometric theory with applications in R. Springer. (Available online at http://personality-project.org/r/book).

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.

See Also

omega, ICLUST, guttman, score.items, cluster.cor

Examples

```
r4 <- sim.congeneric()
alpha(r4)
r9 <- sim.hierarchical()
alpha(r9)
#an example of two independent factors that produce reasonable alphas
#this is a case where alpha is a poor indicator of unidimensionality
two.f <- sim.item(8)
alpha(two.f,keys=c(rep(1,4),rep(-1,4)))
#an example with discrete item responses  -- show the frequencies
items <- sim.congeneric(N=500,short=FALSE,low=-2,high=2,categorical=TRUE) #500 responses
a4 <- alpha(items$observed)  #item response analysis of congeneric measures
a4
#summary just gives Alpha
summary(a4)
```

Bechtoldt *Seven data sets showing a bifactor solution.*

Description

Holzinger-Swineford (1937) introduced the bifactor model of a general factor and uncorrelated group factors. The Holzinger data sets are original 14 * 14 matrix from their paper as well as a 9 *9 matrix used as an example by Joreskog. The Thurstone correlation matrix is a 9 * 9 matrix of correlations of ability items. The Reise data set is 16 * 16 correlation matrix of mental health items. The Bechtholdt data sets are both 17 x 17 correlation matrices of ability tests.

Usage

```
data(Thurstone)
data(Thurstone.33)
data(Holzinger)
data(Holzinger.9)
data(Bechtoldt)
data(Bechtoldt.1)
data(Bechtoldt.2)
data(Reise)
```

Details

Holzinger and Swineford (1937) 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. The sample size is 355.

Another data set from Holzinger (Holzinger.9) represents 9 cognitive abilities (Holzinger, 1939) and is used as an example by Karl Joreskog (2003) for factor analysis by the MINRES algorithm and also appears in the LISREL manual as example NPV.KM.

Another classic data set is the 9 variable Thurstone problem which is discussed in detail by R. P. McDonald (1985, 1999) and and is used as example in the sem package as well as in the PROC CALIS manual for SAS. These nine tests were grouped by Thurstone and Thurstone, 1941 (based on other data) into three factors: Verbal Comprehension, Word Fluency, and Reasoning. The original data came from Thurstone and Thurstone (1941) but were reanalyzed by Bechthold (1961) who broke the data set into two. McDonald, in turn, selected these nine variables from the larger set of 17 found in Bechtoldt.2. The sample size is 213.

Another set of 9 cognitive variables attributed to Thurstone (1933) is the data set of 4,175 students reported by Professor Brigham of Princeton to the College Entrance Examination Board. This set does not show a clear bifactor solution but is included as a demonstration of the differences between a maximimum likelihood factor analysis solution versus a principal axis factor solution.

More recent applications of the bifactor model 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 Health plan customer service (14-16).

The two Bechtoldt data sets are two samples from Thurstone and Thurstone (1941). They include 17 variables, 9 of which were used by McDonald to form the Thurstone data set. The sample sizes are 212 and 213 respectively. The six proposed factors reflect memory, verbal, words, space, number and reasoning with three markers for all expect the rote memory factor. 9 variables from this set appear in the Thurstone data set.

Two more data sets with similar structures are found in the Harman data set.

- Bechtoldt.1: 17 x 17 correlation matrix of ability tests, N = 212.
- Bechtoldt.2: 17 x 17 correlation matrix of ability tests, N = 213.
- Holzinger: 14 x 14 correlation matrix of ability tests, N = 355
- Holzinger.9: 9 x 9 correlation matrix of ability tests, N = 145
- Reise: 16 x 16 correlation matrix of health satisfaction items. N = 35,000
- Thurstone: 9 x 9 correlation matrix of ability tests, N = 213
- Thurstone.33: Another 9 x 9 correlation matrix of ability items, N=4175

Source

Holzinger: Holzinger and Swineford (1937)
Reise: Steve Reise (personal communication)
sem help page (for Thurstone)

References

Bechtoldt, Harold, (1961). An empirical study of the factor analysis stability hypothesis. Psychometrika, 26, 405-432.

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

Holzinger, K., & Swineford, F. (1939). A study in factor analysis: The stability of a bifactor solution. Supplementary Educational Monograph, no. 48. Chicago: University of Chicago Press.

McDonald, Roderick P. (1999) Test theory: A unified treatment. L. Erlbaum Associates. Mahwah, N.J.

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.

Thurstone, Louis Leon (1933) The theory of multiple factors. Edwards Brothers, Inc. Ann Arbor

Thurstone, Louis Leon and Thurstone, Thelma (Gwinn). (1941) Factorial studies of intelligence. The University of Chicago Press. Chicago, Il.

Examples

```
if(!require(GPArotation)) {message("I am sorry, to run omega requires GPArotation") } els
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")
thur.bf <- omega(Thurstone,title="9 variables from Thurstone")
}
```

---

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 2800 subjects are included here as a demonstration set for scale construction, factor analysis, and Item Response Theory analysis. Three additional demographic variables (sex, education, and age) are also included.

Usage

```
data(bfi)
```

Format

A data frame with 2800 observations on the following 28 variables. (The q numbers are the SAPA item numbers).

`A1` Am indifferent to the feelings of others. (q_146)

`A2` Inquire about others' well-being. (q_1162)

`A3` Know how to comfort others. (q_1206)

`A4` Love children. (q_1364)

`A5` Make people feel at ease. (q_1419)

C1 Am exacting in my work. (q_124)

C2 Continue until everything is perfect. (q_530)

C3 Do things according to a plan. (q_619)

C4 Do things in a half-way manner. (q_626)

C5 Waste my time. (q_1949)

E1 Don't talk a lot. (q_712)

E2 Find it difficult to approach others. (q_901)

E3 Know how to captivate people. (q_1205)

E4 Make friends easily. (q_1410)

E5 Take charge. (q_1768)

N1 Get angry easily. (q_952)

N2 Get irritated easily. (q_974)

N3 Have frequent mood swings. (q_1099

N4 Often feel blue. (q_1479)

N5 Panic easily. (q_1505)

O1 Am full of ideas. (q_128)

O2 Avoid difficult reading material.(q_316)

O3 Carry the conversation to a higher level. (q_492)

O4 Spend time reflecting on things. (q_1738)

O5 Will not probe deeply into a subject. (q_1964)

gender Males = 1, Females =2

education 1 = HS, 2 = finished HS, 3 = some college, 4 = college graduate 5 = graduate degree

age age in years

Details

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

These five factors are a useful example of using `irt.fa` to do Item Response Theory based latent factor analysis of the `polychoric` correlation matrix. The endorsement plots for each item, as well as the item information functions reveal that the items differ in their quality.

The item data were collected using a 6 point response scale: 1 Very Inaccurate 2 Moderately Inaccurate 3 Slightly Inaccurate 4 Slightly Accurate 5Moderately Accurate 6 Very Accurate

as part of the Synthetic Apeture Personality Assessment (SAPA) project. To see an example of the data collection technique, visit `http://test.personality-project.org`. The items given were sampled from the International Personality Item Pool of Lewis Goldberg using the sampling technique of SAPA. This is a sample data set taken from the much larger SAPA data bank.

Source

The items are from the ipip (Goldberg, 1999). The data are from the SAPA project (Revelle, Wilt and Rosenthal, 2010) , collected Spring, 2010.

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. (2010) 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.

See Also

bi.bars to show the data by age and gender, irt.fa for item factor analysis applying the irt model.

Examples

```
data(bfi)
describe(bfi)
 keys.list <- list(Agree=c(-1,2:5),Conscientious=c(6:8,-9,-10),Extraversion=c(-11,-12,13:
 keys <- make.keys(28,keys.list,item.labels=colnames(bfi))
 score.items(keys,bfi)
 scores <- score.items(keys[1:27,],bfi[1:27]) #don't score age
 scores
```

---

bi.bars *Draw pairs of bargraphs based on two groups*

Description

When showing e.g., age or education distributions for two groups, it is convenient to plot them back to back. bi.bars will do so.

Usage

```
bi.bars(x,grp,horiz,color,...)
```

Arguments

x	The data to be drawn
grp	a grouping variable.
horiz	horizontal (default) or vertical bars
color	colors for the two groups – defaults to blue and red
...	Further parameters to pass to the graphing program

Details

A trivial, if useful, function to draw back to back histograms/barplots. One for each group.

Value

a graphic

Author(s)

William Revelle

Examples

```
data(bfi)
with(bfi,{bi.bars(age,gender,ylab="Age",main="Age by males and females")
       bi.bars(education,gender,xlab="Education",main="Education by gender",horiz=FALSE)}
```

---

biplot.psych *Draw biplots of factor or component scores by factor or component loadings*

Description

Applies the biplot function to the output of fa or principal. Will plot factor scores and factor loadings in the same graph. If the number of factors > 2, then all pairs of factors are plotted. Factor score histograms are plotted on the diagonal. The input is the resulting object from fa or principal with the scores=TRUE option.

Usage

```
biplot.psych(x, labels=NULL,cex=c(.75,1),main="Biplot",hist.col="cyan",xlim=c(-3
```

Arguments

x	The output from fa or principal with the scores=TRUE option
labels	if NULL, draw the points with small o. To identify the data points, specify labels= 1:n where n is the number of observations, or labels =rownames(data) where data was the data set analyzed by the factor analysis.
cex	plot size of the data points and of the factor labels
main	A main title for a two factor biplot
hist.col	If plotting more than two factors, the color of the histogram of the factor scores
xlim	Defaults to plus/minus three sigma
ylim	Defaults to plus/minus three sigma
...	more options for graphics

Details

Uses the generic biplot function to take the output of a factor analysis fa or principal components analysis principal and plot the factor/component scores along with the factor/component loadings.

Author(s)

William Revelle

See Also

fa, principal, factor.plot, pairs.panels

Examples

```
data(USArrests)
fa2 <- fa(USArrests,2,scores=TRUE)
biplot(fa2,labels=rownames(USArrests))
data(bfi)
fa3 <- fa(bfi[1:200,1:15],3,scores=TRUE)
biplot(fa3)
```

---

block.random *Create a block randomized structure for n independent variables*

Description

Random assignment of n subjects with an equal number in all of N conditions may done by block randomization, where the block size is the number of experimental conditions. The number of Independent Variables and the number of levels in each IV are specified as input. The output is a the block randomized design.

Usage

```
block.random(n, ncond = NULL)
```

Arguments

n	The number of subjects to randomize. Must be a multiple of the number of experimental conditions
ncond	The number of conditions for each IV. Defaults to 2 levels for one IV. If more than one IV, specify as a vector. If names are provided, they are used, otherwise the IVs are labeled as IV1 ... IVn

Value

blocks	A matrix of subject numbers, block number, and randomized levels for each IV

Note

Prepared for a course on Research Methods in Psychology http://personality-project.org/revelle/syllabi/205/205.syllabus.html

Author(s)

William Revelle

Examples

```
br <- block.random(n=24,c(2,3))
pairs.panels(br)
br <- block.random(96,c(time=4,drug=3,sex=2))
pairs.panels(br)
```

blot *Bond's Logical Operations Test - BLOT*

Description

35 items for 150 subjects from Bond's Logical Operations Test. A good example of Item Response Theory analysis using the Rasch model. One parameter (Rasch) analysis and two parameter IRT analyses produce somewhat different results.

Usage

```
data(blot)
```

Format

A data frame with 150 observations on 35 variables. The BLOT was developed as a paper and pencil test for children to measure Logical Thinking as discussed by Piaget and Inhelder.

Details

Bond and Fox apply Rasch modeling to a variety of data sets. This one, Bond's Logical Operations Test, is used as an example of Rasch modeling for dichotomous items. In their text (p 56), Bond and Fox report the results using WINSTEPS. Those results are consistent (up to a scaling parameter) with those found by the rasch function in the ltm package. The WINSTEPS seem to produce difficulty estimates with a mean item difficulty of 0, whereas rasch from ltm has a mean difficulty of -1.52. In addition, rasch seems to reverse the signs of the difficulty estimates when reporting the coefficients and is effectively reporting "easiness".

However, when using a two parameter model, one of the items (V12) behaves very differently.

This data set is useful when comparing 1PL, 2PL and 2PN IRT models.

Source

The data are taken (with kind permission from Trevor Bond) from the webpage http://homes.jcu.edu.au/~edtgb/book/data/Bond87.txt and read using read.fwf.

References

T.G. Bond. BLOT:Bond's Logical Operations Test. Townsville, Australia: James Cook University. (Original work published 1976), 1995.

T. Bond and C. Fox. (2007) Applying the Rasch model: Fundamental measurement in the human sciences. Lawrence Erlbaum, Mahwah, NJ, US, 2 edition.

See Also

See also the irt.fa and associated plot functions.

Examples

```
data(blot)
#not run
#library(ltm)
#bblot.rasch <- rasch(blot, constraint = cbind(ncol(blot) + 1, 1))  #a 1PL model
#blot.2pl <- ltm(blot~z1)  #a 2PL model
#do the same thing with functions in psych
#blot.fa <- irt.fa(blot)  # a 2PN model
#plot(blot.fa)
```

`bock` *Bock and Liberman (1970) data set of 1000 observations of the LSAT*

Description

An example data set used by McDonald (1999) as well as other discussions of Item Response Theory makes use of a data table on 10 items (two sets of 5) from the Law School Admissions Test (LSAT). Included in this data set is the original table as well as the reponses for 1000 subjects on the first set (Figure Classification) and second set (Debate).

Usage

```
data(bock)
```

Format

A data frame with 32 observations on the following 8 variables.

`index` 32 response patterns

`Q1` Responses to item 1

`Q2` Responses to item 2

`Q3` Responses to item 3

`Q4` Responses to item 4

`Q5` Responses to item 5

`Ob6` count of observations for the section 6 test

`Ob7` count of observations for the section 7 test

Two other data sets are derived from the bock dataset. These are converted using the `table2df` function.

lsat6 reponses to 5 items for 1000 subjects on section 6

lsat7 reponses to 5 items for 1000 subjects on section 7

Details

The lsat6 data set is analyzed in the ltm package as well as by McDonald (1999). lsat7 is another 1000 subjects on part 7 of the LSAT. Both sets are described by Bock and Lieberman (1970). Both sets are useful examples of testing out IRT procedures and showing the use of `tetrachoric` correlations and item factor analysis using the `irt.fa` function.

Source

R. Darrell Bock and M. Lieberman (1970). Fitting a response model for dichotomously scored items. Psychometrika, 35(2):179-197.

References

R.P. McDonald. Test theory: A unified treatment. L. Erlbaum Associates, Mahwah, N.J., 1999.

Examples

```
data(bock)
responses <- table2df(bock.table[,2:6],count=bock.table[,7],labs= paste("lsat6.",1:5,sep=
describe(responses)
## maybe str(bock.table) ; plot(bock.table) ...
```

`burt` *11 emotional variables from Burt (1915)*

Description

Cyril Burt reported an early factor analysis with a circumplex structure of 11 emotional variables in 1915. 8 of these were subsequently used by Harman in his text on factor analysis. Unfortunately, it seems as if Burt made a mistake for the matrix is not positive definite. With one change from .87 to .81 the matrix is positive definite.

Usage

```
data(burt)
```

Format

A correlation matrix based upon 172 "normal school age children aged 9-12".

Sociality Sociality

Sorrow Sorrow

Tenderness Tenderness

Joy Joy

Wonder Wonder

Elation Elation

Disgust Disgust

Anger Anger

Sex Sex

Fear Fear

Subjection Subjection

Details

The Burt data set is interesting for several reasons. It seems to be an early example of the organizaton of emotions into an affective circumplex, a subset of it has been used for factor analysis examples (see Harman.Burt, and it is an example of how typos affect data. The original data matrix has one negative eigenvalue. With the replacement of the correlation between Sorrow and Tenderness from .87 to .81, the matrix is positive definite.

Source

(retrieved from the web at http://www.biodiversitylibrary.org/item/95822#790) Following a suggestion by Jan DeLeeuw.

References

Burt, C.General and Specific Factors underlying the Primary Emotions. Reports of the British Association for the Advancement of Science, 85th meeting, held in Manchester, September 7-11, 1915. London, John Murray, 1916, p. 694-696 (retrieved from the web at http://www.biodiversitylibrary.org/item/95822#790)

See Also

Harman.Burt in the Harman dataset.

Examples

```
data(burt)
eigen(burt)$values  #one is negative!
burt.new <- burt
burt.new[2,3] <- burt.new[3,2] <- .80
eigen(burt.new)$values  #all are positive
```

---

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 nonzero 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

To understand the results of the circ.tests it it best to compare it to simulated values. Thus, see circ.simulation, sim.circ

Examples

```
circ.data <- circ.sim(24,500)
circ.fa <- factor.pa(circ.data,2)
plot(circ.fa,title="Circumplex Structure")
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,title="Simple Structure")
st <- circ.tests(simp.fa)
res <- rbind(ct[1:4],st[1:4])
rownames(res) <- c("circumplex","Simple")
print(res,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]
#not run
#an overlay map can be added if the package maps is available
#
#
#libary(maps)
#map("usa")
#title("MultiDimensional Scaling of US cities")
#points(city.location)

plot(city.location, xlab="Dimension 1", ylab="Dimension 2",main ="Multidimensional scalin
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. Similar functionality to score.items which also gives item by cluster correlations.

Usage

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

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
SMC	Should squared multiple correlations be used as communality estimates for the correlation matrix?

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 or from the correlation matrix using score.items which is probably preferred.

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.
G6	Guttman’s Lambda 6 reliability estimate is based upon the smcs for each item in a scale. G6 uses the smc based upon the entire item domain.
av.r	The average inter item correlation within a scale
size	How many items are in each cluster?

Note

See SAPA Revelle, W., Wilt, J., and Rosenthal, A. (2010) 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)
```

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?

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,SMC=TRUE)
```

Arguments

`keys`	Cluster keys: a matrix of -1,0,1 cluster weights
`r.mat`	A correlation matrix
`correct`	Correct for reliability
`SMC`	Use the squared multiple correlation as a communality estimate, otherwise use the greatest correlation for each variable

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
`G6`	G6* Modified estimated of Guttman Lambda 6
`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

```
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)
```

---

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

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).

Both of these functions may be called directly or by calling the generic plot function. (see example).

Value

Graphical output is presented.

Author(s)

William Revelle

See Also

ICLUST, ICLUST.graph, fa.graph, plot.psych

Examples

```
circ.data <- circ.sim(24,500)
circ.fa <- fa(circ.data,2)
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.

May also be used to take the output of an ICLUST analysis and find a keys matrix. (By doing a call to the factor2cluster function.

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, factor2cluster, make.keys

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)
```

---

cohen.kappa *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.

weighted.kappa 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

```
cohen.kappa(x, w=NULL,n.obs=NULL,alpha=.05)
wkappa(x, w = NULL)    #deprectated
```

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 0 (on the diagonal) and (distance from diagonal) off the diagonal)^2.
n.obs	Number of observations (if input is a square matrix.
alpha	Probability level for confidence intervals

Details

When cateogorical judgments are made with two cateories, a measure of relationship is the phi coefficient. However, 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) or to categorize the stages of a disease. 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).

Kappa considers the matches on the main diagonal. A penalty function (weight) may be applied to the off diagonal matches. If the weights increase by the square of the distance from the diagonal, weighted kappa is similar to an Intra Class Correlation (`ICC`).

Derivations of weighted kappa are sometimes expressed in terms of similarities, and sometimes in terms of dissimilarities. In the latter case, the weights on the diagonal are 1 and the weights off the diagonal are less than one. In this, if the weights are 1 - squared distance from the diagonal / k, then the result is similar to the ICC (for any positive k).

cohen.kappa may use either similarity weighting (diagonal = 0) or dissimilarity weighting (diagonal = 1) in order to match various published examples.

The input may be a two column data.frame or matrix with columns representing the two judges and rows the subjects being rated. Alternatively, the input may be a square n x n matrix of counts or proportion of matches. If proportions are used, it is necessary to specify the number of observations (n.obs) in order to correctly find the confidence intervals.

The confidence intervals are based upon the variance estimates discussed by Fleiss, Cohen, and Everitt who corrected the formulae of Cohen (1968) and Blashfield.

Value

`kappa`	Unweighted kappa
`weighted.kappa`	The default weights are quadratric.
`var.kappa`	Variance of kappa
`var.weighted`	Variance of weighted kappa
`n.obs`	number of observations
`weight`	The weights used in the estimation of weighted kappa
`confid`	The alpha/2 confidence intervals for unweighted and weighted kappa
`plevel`	The alpha level used in determining the confidence limits

Note

As is true of many R functions, there are alternatives in other packages. The Kappa function in the vcd package estimates unweighted and weighted kappa and reports the variance of the estimate. The input is a square matrix. The ckappa and wkappa functions in the psy package take raw data matrices.

To avoid confusion with Kappa (from vcd) or the kappa function from base, the function was originally named wkappa. With additional features modified from psy::ckappa to allow input with a different number of categories, the function has been renamed cohen.kappa.

Unfortunately, to make it more confusing, the weights described by Cohen are a function of the reciprocals of those discucssed by Fleiss and Cohen. The cohen.kappa function uses the appropriate formula for Cohen or Fleiss-Cohen weights.

Author(s)

William Revelle

References

Banerjee, M., Capozzoli, M., McSweeney, L and Sinha, D. (1999) Beyond Kappa: A review of interrater agreement measures The Canadian Journal of Statistics / La Revue Canadienne de Statistique, 27, 3-23

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.

Fleiss, J. L., Cohen, J. and Everitt, B.S. (1969) Large sample standard errors of kappa and weighted kappa. Psychological Bulletin, 72, 332-327.

Zwick, R. (1988) Another look at interrater agreement. Psychological Bulletin, 103, 374 - 378.

Examples

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

#cohen.weights  weight differences
cohen.weights <- matrix(c(
0,1,3,
1,0,6,
3,6,0),ncol=3)


cohen.kappa(cohen,cohen.weights,n.obs=200)
#cohen reports .492 and .348

#another set of weights
#what if the weights are non-symmetric
wc <- matrix(c(
0,1,4,
1,0,6,
2,2,0),ncol=3,byrow=TRUE)
cohen.kappa(cohen,wc)
#Cohen reports kw = .353

cohen.kappa(cohen,n.obs=200)  #this uses the squared weights

fleiss.cohen <- 1 - cohen.weights/9
cohen.kappa(cohen,fleiss.cohen,n.obs=200)

#however, Fleiss, Cohen and Everitt weight similarities
fleiss <- matrix(c(
106, 10,4,
22,28, 10,
2, 12,  6),ncol=3,byrow=TRUE)
```

```
#Fleiss weights the similarities
weights <- matrix(c(
 1.0000, 0.0000, 0.4444,
 0.0000, 1.0000, 0.6666,
 0.4444, 0.6666, 1.0000),ncol=3)

 cohen.kappa(fleiss,weights,n.obs=200)

 #data may be a 2 column matrix
 #compare weighted and unweighted
 #also look at the ICC for this data set.
 twins <- matrix(c(
    1, 2,
    2, 3,
    3, 4,
    5, 6,
    6, 7), ncol=2,byrow=TRUE)
  cohen.kappa(twins)

#data may be explicitly categorical
x <- c("red","yellow","blue","red")
y <- c("red",  "blue", "blue" ,"red")
xy.df <- data.frame(x,y)
ck <- cohen.kappa(xy.df)
ck
ck$agree

#finally, input can be a data.frame of ratings from more than two raters
ratings <- matrix(rep(1:5,4),ncol=4)
ratings[1,2] <- ratings[2,3] <- ratings[3,4] <- NA
ratings[2,1] <- ratings[3,2] <- ratings[4,3] <- 1
cohen.kappa(ratings)
```

`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

Author(s)

William Revelle

See Also

phi, Yule

Examples

```
comorbidity(.2,.15,.1,c("Anxiety","Depression"))
```

---

cor.plot *Create an image plot for a correlation or factor matrix*

Description

Correlation matrices may be shown graphically by using the image function to emphasize structure. This is a particularly useful tool for showing the structure of correlation matrices with a clear structure. Partially meant for the pedagogical value of the graphic for teaching or discussing factor analysis and other multivariate techniques.

Usage

```
cor.plot(r,colors=TRUE, n=51,main=NULL,zlim=c(-1,1),show.legend=TRUE,labels=NULL
```

Arguments

r	A correlation matrix or the output of factor.pa, factor.minres or omega.
colors	Defaults to TRUE and colors use colors from the colorRampPalette from red through white to blue, but colors=FALSE will use a grey scale
n	The number of levels of shading to use. Defaults to 51
main	A title. Defaults to "correlation plot"
zlim	The range of values to color – defaults to -1 to 1
show.legend	A legend (key) to the colors is shown on the right hand side
labels	if NULL, use column and row names, otherwise use labels
n.legend	How many categories should be labelled in the legend?
...	Other parameters for axis (e.g., cex.axis to change the font size)

Details

When summarizing the correlations of large data bases or when teaching about factor analysis or cluster analysis, it is useful to graphically display the structure of correlation matrices. This is a simple graphical display using the image function.

The difference of mat.plot with a regular image plot is that the primary diagonal goes from the top left to the lower right. -1 to 1 and the color choice is more reasonable. Setting it as c(0,1) will lead to negative correlations treated as zero. This is advantageous when showing general factor structures, because it makes the 0 white.

The default shows a legend for the color coding on the right hand side of the figure.

Inspired, in part, by a paper by S. Dray (2008) on the number of components problem.

Modified following suggestions by David Condon and Josh Wilt to use a more meaningful color choice ranging from dark red (-1) through white (0) to dark blue (1).

Author(s)

William Revelle

References

Dray, Stephane (2008) On the number of principal components: A test of dimensionality based on measurements of similarity between matrices. Computational Statistics \& Data Analysis. 52, 4, 2228-2237.

See Also

fa, mat.sort

Examples

```
cor.plot(Thurstone,main="9 cognitive variables from Thurstone") #just blue implies positi
cor.plot(Thurstone, zlim=c(0,1),main="9 cognitive variables from Thurstone")
cor.plot(mat.sort(Thurstone),TRUE,zlim=c(0,1), main="9 cognitive variables from Thurstone
simp <- sim.circ(24)
cor.plot(cor(simp),main="24 variables in a circumplex")
```

---

corr.test *Find the correlations, sample sizes, and probability values between elements of a matrix or data.frame.*

Description

Although the cor function finds the correlations for a matrix, it does not report probability values. corr.test uses cor to find the correlations for either complete or pairwise data and reports the sample sizes and probability values as well.

Usage

```
corr.test(x, y = NULL, use = "pairwise",method="pearson")
```

Arguments

x	A matrix or dataframe
y	A second matrix or dataframe with the same number of rows as x
use	use="pairwise" is the default value and will do pairwise deletion of cases. use="complete" will select just complete cases.
method	method="pearson" is the default value. The alternatives to be passed to cor are "spearman" and "kendall"

Details

corr.test uses the cor function to find the correlations, and then applies a t-test to the individual correlations using the formula

$$t = \frac{r * \sqrt{(n-2)}}{\sqrt{(1-r^2)}}$$

Value

r	The matrix of correlations
n	Number of cases per correlation
t	value of t-test for each correlation
p	two tailed probability of t for each correlation

See Also

cor.test for tests of a single correlation, Hmisc::rcorr for an equivalant function, r.test to test the difference between correlations, and cortest.mat to test for equality of two correlation matrices.

Examples

```
data(sat.act)
corr.test(sat.act)
```

---

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(a1)
alpha2 <- alpha(a2)
x <- matrix(c(x1,x2),ncol=2)
x.cor <- cor(x)
alpha <- c(alpha1$total$raw_alpha,alpha2$total$raw_alpha)
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(angle,x=NULL,code=NULL,period=24,plot=FALSE,opti=FALSE)
circadian.mean(angle, hours=TRUE)
circadian.cor(angle, hours=TRUE)
circadian.linear.cor(angle,x,hours=TRUE)
```

Arguments

angle	A data frame or matrix of observed values with the time of day as the first value (unless specified in code) angle can be specified either as hours or as radians)
code	A subject identification variable
period	Although time of day is assumed to have a 24 hour rhythm, other rhythms may be fit.
plot	if TRUE, then plot the first variable (angle)
opti	opti=TRUE: iterative optimization (slow) or opti=FALSE: linear fitting (fast)
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 generalization 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) give a 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 analysis of data from various studies of mood over the day.

The cosinor function will either iteratively fit cosines of the angle to the observed data (opti=TRUE) or use the circular by linear regression to estimate the best fitting phase angle. If cos.t <- cos(time) and sin.t = sin(time) (expressed in hours), then beta.c and beta.s may be found by regression and the phase is $sign(beta.c) * acos(beta.c/\sqrt{(beta.c^2 + beta.s^2)}) * 12/pi$

Simulations (see examples) suggest that with incomplete times, perhaps the optimization procedure yields slightly better fits with the correct phase than does the linear model, but the differences are very small. In the presence of noisey data, these advantages seem to reverse. The recommendation thus seems to be to use the linear model approach (the default).

Value

`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.

Examples

```
time <- seq(1:24)
pure <- matrix(time,24,18)
pure <- cos((pure + col(pure))*pi/12)
matplot(pure,type="l",main="Pure circadian arousal rhythms",xlab="time of day",ylab="Arou
```

```
p <- cosinor(time,pure)
#set.seed(42)
noisy <- pure + rnorm(24*18)
n <- cosinor(time,noisy)
#small.pure <- pure[c(6:18),]
small.pure <- pure[c(8,11,14,17,20,23),]
#small.noisy <- noisy[c(6:18),]
small.noisy <- noisy[c(8,11,14,17,20,23),]
matplot(small.noisy,type="l",main="Noisy circadian arousal rhythms",xlab="time of day",yl
#sp <- cosinor(time[c(6:18)],small.pure)  #linear fit
sp <- cosinor(time[c(8,11,14,17,20,23)],small.pure)
spo <- cosinor(time[c(8,11,14,17,20,23)],small.pure,opti=TRUE) #iterative fit
sn <- cosinor(time[c(8,11,14,17,20,23)],small.noisy) #linear
sno <- cosinor(time[c(8,11,14,17,20,23)],small.noisy,opti=TRUE) #iterative
sum.df <- data.frame(pure=p,noisy = n, small=sp,small.noise = sn, small.opt=spo,small.noi
round(sum.df,2)
round(circadian.cor(sum.df[,c(1,3,5,7,9,11)]),2)  #compare alternatives
round(cor(sum.df[,c(2,4,6,8,10,12)]),2)
```

`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)
```

`cta` *Simulate the C(ues) T(endency) A(ction) model of motivation*

Description

Dynamic motivational models such as the Dynamics of Action (Atkinson and Birch, 1970, Revelle, 1986) may be reparameterized as a simple pair of differential (matrix) equations (Revelle, 1986, 2008). This function simulates the dynamic aspects of the CTA.

Usage

```
cta(n = 3, t = 5000, cues = NULL, act = NULL, inhibit = NULL, consume = NULL, te
```

Arguments

`n`	number of actions to simuate
`t`	length of time to simulate
`cues`	a vector of cue strengths
`act`	matrix of associations between cues and action tendencies
`inhibit`	inhibition matrix
`consume`	Consummation matrix
`ten`	Initial values of action tendencies
`type`	show actions, tendencies, both, or state diagrams
`fast`	display every fast time (skips
`compare`	Compare?

Details

A very thorough discussion of the CTA model is available from Revelle (2008).

Value

graphical output

`cues`	echo back the cue input
`inhibition`	echo back the inhibitory matrix
`time`	time spent in each activity
`frequency`	Frequency of each activity
`ten`	final tension values
`act`	final action values

Author(s)

William Revelle

References

Atkinson, John W. and Birch, David (1970) The dynamics of action. John Wiley, New York, N.Y.

Revelle, William (1986) Motivation and efficiency of cognitive performance in Brown, Donald R. and Veroff, Joe (ed). Frontiers of Motivational Psychology: Essays in honor of J. W. Atkinson. Springer.

Revelle, W. (2008) Cues, Tendencies and Actions. The Dynamics of Action revisted. http://personality-project.org/revelle/publications/cta.pdf

Examples

```
#not run
#cta()   #default values, running over time
#cta(type="state") #default values, in a state space  of tendency 1 versus tendency 2
```

---

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`, `ellipses`, `heights`, `peas`,`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-relat
```

---

cushny *A data set from Cushny and Peebles (1905) on the effect of three drugs on hours of sleep, used by Student (1908)*

Description

The classic data set used by Gossett (publishing as Student) for the introduction of the t-test. The design was a within subjects study with hours of sleep in a control condition compared to those in 3 drug conditions. Drug1 was 06mg of L Hscyamine, Drug 2L and Drug2R were .6 mg of Left and Right isomers of Hyoscine. The detal1, delta2L and delta2R are changes from the baseline control.

Usage

```
data(cushny)
```

Format

A data frame with 10 observations on the following 7 variables.

`Control` Hours of sleep in a control condition

`drug1` Hours of sleep in Drug condition 1

`drug2L` Hours of sleep in Drug condition 2

drug2R Hours of sleep in Drug condition 3 (an isomer of the drug in condition 2

delta1 Change from control, drug 1

delta2L Change from control, drug 2L

delta2R Change from control, drug 2R

Details

The original analysis by Student is used as an example for the t-test function, both as a paired t-test and a two group t-test. The data are also useful for a repeated measures analysis of variance.

Source

Cushny, A.R. and Peebles, A.R. (1905) The action of optical isomers: II hyoscines. The Journal of Physiology 32, 501-510.

Student (1908) The probable error of the mean. Biometrika, 6 (1) , 1-25.

References

See also the data set sleep and the examples for the t.test

Examples

```
data(cushny)
with(cushny, t.test(drug1,drug2L,paired=TRUE)) #within subjects

error.bars(cushny[1:4],within=TRUE,ylab="Hours of sleep",xlab="Drug condition",main="95%
```

---

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, na.rm = TRUE, interp=FALSE,skew = TRUE, ranges = TRUE,trim=.1)
```

Arguments

x	A data frame or matrix
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 alphanumeric data). Variables that are categorical or logical are converted to numeric and then described. These variables are marked with an * in the row name.

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

```
data(sat.act)
describe(sat.act)

describe(sat.act,skew=FALSE)
```

---

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. Partly 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. In the case of matrix output with multiple grouping variables, the grouping variable values are added to the output.

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$education,mat=TRUE) #matrix  (data.frame) outp
des.mat <- describe.by(sat.act$age,list(sat.act$education,sat.act$gender),mat=TRUE)  #mat
```

---

diagram *Helper functions for drawing path model diagrams*

Description

Path models are used to describe structural equation models or cluster analytic output. These functions provide the primitives for drawing path models. Used as a substitute for some of the functionality of Rgraphviz.

Usage

```
diagram(fit,...)
dia.rect(x, y = NULL, labels = NULL,  cex = 1,  xlim = c(0, 1), ylim = c(0, 1),
 dia.ellipse(x, y = NULL, labels = NULL, cex=1,e.size=.05, xlim=c(0,1), ylim=c(0
 dia.triangle(x, y = NULL, labels =NULL,  cex = 1, xlim=c(0,1),ylim=c(0,1),...)
dia.ellipse1(x,y,e.size=.05,xlim=c(0,1),ylim=c(0,1),...)
dia.shape(x, y = NULL, labels = NULL, cex = 1, e.size=.05, xlim=c(0,1), ylim=c(0
dia.arrow(from,to,labels=NULL,scale=1,cex=1,...)
dia.curve(from,to,labels=NULL,scale=1,...)
dia.curved.arrow(from,to,labels=NULL,scale=1,...)
dia.self(location,labels=NULL,scale=.8,side=2,...)
```

Arguments

fit	The results from a factor analysis fa, components analysis principal, omega reliability analysis, omega, cluster analysis iclust or confirmatory factor analysis, cfa, or structural equation model,sem, using the lavaan package.
x	x coordinate of a rectangle or ellipse
y	y coordinate of a rectangle or ellipse
e.size	The size of the ellipse (scaled by the number of variables
labels	Text to insert in rectangle, ellipse, or arrow
cex	adjust the text size
scale	modifies size of rectangle and ellipse as well as the curvature of curves. (For curvature, positive numbers are concave down and to the left
from	arrows and curves go from
to	arrows and curves go to
location	where is the rectangle?
shape	Which shape to draw
xlim	default ranges
ylim	default ranges
side	Which side of boxes should errors appear
...	Most graphic parameters may be passed here

Details

The diagram function calls fa.diagram, omega.diagram, ICLUST.diagram or lavaan.diagram depending upon the class of the fit input. See those functions for particular parameter values.

The remaining functions are the graphic primitives used by fa.diagram, structure.diagram, omega.diagram, and ICLUST.diagram.

They create rectangles, ellipses or triangles surrounding text, connect them to straight or curved arrows, and can draw an arrow from and to the same rectangle.

Each shape (ellipse, rectangle or triangle) has a left, right, top and bottom and center coordinate that may be used to connect the arrows.

Curves are double-headed arrows.

The helper functions were developed to get around the infelicities associated with trying to install Rgraphviz and graphviz.

Better documentation will be added as these functions get improved. Currently the helper functions are just a work around for Rgraphviz.

Value

Graphic output

Author(s)

William Revelle

See Also

The diagram functions that use the dia functions: fa.diagram, structure.diagram, omega.diagram, and ICLUST.diagram.

Examples

```
#first, show the primitives
xlim=c(-2,10)
ylim=c(0,10)
plot(NA,xlim=xlim,ylim=ylim,main="Demonstration of diagram functions",axes=FALSE,xlab="",
ul <- dia.rect(1,9,labels="upper left",xlim=xlim,ylim=ylim)
ml <- dia.rect(1,6,"middle left",xlim=xlim,ylim=ylim)
ll <- dia.rect(1,3,labels="lower left",xlim=xlim,ylim=ylim)
bl <- dia.rect(1,1,"bottom left",xlim=xlim,ylim=ylim)
lr <- dia.ellipse(7,3,"lower right",xlim=xlim,ylim=ylim,e.size=.07)
ur <- dia.ellipse(7,9,"upper right",xlim=xlim,ylim=ylim,e.size=.07)
mr <- dia.ellipse(7,6,"middle right",xlim=xlim,ylim=ylim,e.size=.07)
lm <- dia.triangle(4,1,"Lower Middle",xlim=xlim,ylim=ylim)
br <- dia.rect(9,1,"bottom right",xlim=xlim,ylim=ylim)
dia.curve(from=ul$left,to=bl$left,"double headed",scale=-1)

dia.arrow(from=lr,to=ul,labels="right to left")
dia.arrow(from=ul,to=ur,labels="left to right")
dia.curved.arrow(from=lr,to=ll,labels ="right to left")
dia.curved.arrow(to=ur,from=ul,labels ="left to right")
dia.curve(ll$top,ul$bottom,"right")  #for rectangles, specify where to point

dia.curve(ll$top,ul$bottom,"left",scale=-1)  #for rectangles, specify where to point
dia.curve(mr,ur,"up")  #but for ellipses, you may just point to it.
dia.curve(mr,lr,"down")
dia.curve(mr,ur,"up")
dia.curved.arrow(mr,ur,"up")  #but for ellipses, you may just point to it.
dia.curved.arrow(mr,lr,"down")  #but for ellipses, you may just point to it.

dia.curved.arrow(ur$right,mr$right,"3")
dia.curve(ml,mr,"across")
dia.curve(ur,lr,"top down")
dia.curved.arrow(br$top,lr$bottom,"up")
dia.curved.arrow(bl,br,"left to right")
dia.curved.arrow(br,bl,"right to left",scale=-1)
dia.arrow(bl,ll$bottom)
dia.curved.arrow(ml,ll$right)
dia.curved.arrow(mr,lr$top)

#now, put them together in a factor analysis diagram
v9 <- sim.hierarchical()
f3 <- fa(v9,3,rotate="cluster")
fa.diagram(f3,error=TRUE,side=3)
```

`draw.tetra` *Draw a correlation ellipse and two normal curves to demonstrate tetrachoric correlation*

Description

A graphic of a correlation ellipse divided into 4 regions based upon x and y cutpoints on two normal distributions. This is also an example of using the layout function.

Usage

```
draw.tetra(r, t1, t2,shade=TRUE)
```

Arguments

r	the underlying Pearson correlation defines the shape of the ellipse
t1	X is cut at tau
t2	Y is cut at Tau
shade	shade the diagram (default is TRUE)

Details

A graphic demonstration of the tetrachoric correlation. Used for teaching purposes. The default values are for a correlation of .5 with cuts at 1 and 1. Any other values are possible. The code is also a demonstration of how to use the layout function for complex graphics using base graphics.

Author(s)

William Revelle

See Also

tetrachoric to find tetrachoric correlations, irt.fa and fa.poly to use them in factor analyses, scatter.hist to show correlations and histograms.

Examples

```
if(require(mvtnorm)) {
draw.tetra(.5,1,1)
draw.tetra(.8,2,1)} else {print("draw.tetra requires the mvtnorm package")}
```

Dwyer *8 cognitive variables used by Dwyer for an example.*

Description

Dwyer (1937) introduced a technique for factor extension and used 8 cognitive variables from Thurstone. This is the example data set used in his paper.

Usage

```
data(Dwyer)
```

Format

The format is: num [1:8, 1:8] 1 0.58 -0.28 0.01 0.36 0.38 0.61 0.15 0.58 1 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:8] "V1" "V2" "V3" "V4" ... ..$: chr [1:8] "V1" "V2" "V3" "V4" ...

Source

Data matrix retyped from the original publication.

References

Dwyer, Paul S. (1937), The determination of the factor loadings of a given test from the known factor loadings of other tests. Psychometrika, 3, 173-178

Examples

```
data(Dwyer)
Ro <- Dwyer[1:7,1:7]
Roe <- Dwyer[1:7,8]
fo <- fa(Ro,2,rotate="none")
fa.extension(Roe,fo)
```

---

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. The minkowski function is included as a generalized ellipse.

Usage

```
ellipses(x, y = NULL, add = FALSE, smooth=TRUE, lm=FALSE,data=TRUE, n = 2,span=2
minkowski(r=2,add=FALSE,main=NULL,xl=1,yl=1)
```

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
r	r=1 draws a city block, r=2 is a Euclidean circle, r > 2 tends towards a square
main	title to use when drawing Minkowski circles
xl	stretch the x axis
yl	stretch the y axis

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
data(sat.act)
ellipses(sat.act)  #shows the pairs.panels ellipses
minkowski(2,main="Minkowski circles")
minkowski(1,TRUE)
minkowski(4,TRUE)
```

---

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` *Plot means and confidence intervals*

Description

One of the many functions in R to plot means and confidence intervals. Can be done using barplots if desired. Can also be combined with such functions as boxplot to summarize distributions. Means and standard errors are calculated from the raw data using `describe`. Alternatively, plots of means +/- one standard deviation may be drawn.

Usage

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

Arguments

`x`	A data frame or matrix of raw data
`stats`	Alternatively, a data.frame of descriptive stats from (e.g., describe)
`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
`sd`	if TRUE, draw one standard deviation instead of standard errors at the alpha level
`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?
`arrow.col`	What color should the error bars be?
`add`	add=FALSE, new plot, add=TRUE, just points and error bars
`bars`	bars=TRUE will draw a bar graph if you really want to do that
`within`	should the error variance of a variable be corrected by 1-SMC?
`...`	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.

If within=TRUE, the error bars are corrected for the correlation with the other variables by reducing the variance by a factor of (1-smc). This allows for comparisons between variables.

The error bars are normally calculated from the data using the describe function. If, alternatively, a matrix of statistics is provided with column headings of values, means, and se, then those values will be used for the plot (using the stats option). However, in this case, the error bars will be one s.e. rather than a function of the alpha level.

If sd is TRUE, then the error bars will represent one standard deviation from the mean rather than be a function of alpha and the standard errors.

Value

Graphic output showing the means + x

These confidence regions are based upon normal theory and do not take into account any skew in the variables. More accurate confidence intervals could be found by resampling.

Author(s)

William Revelle

See Also

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

In addition, as pointed out by Jim Lemon on the R-help news group, error bars or confidence intervals may be drawn using

function	package
bar.err	(agricolae)
plotCI	(gplots)
xYplot	(Hmisc)
dispersion	(plotrix)
plotCI	(plotrix)

For advice why not to draw bar graphs with error bars, see http://biostat.mc.vanderbilt.edu/wiki/Main/DynamitePlots

Examples

```
x <- replicate(20,rnorm(50))
boxplot(x,notch=TRUE,main="Notched boxplot with error bars")
error.bars(x,add=TRUE)
abline(h=0)

error.bars(attitude,alpha=.5,main="50 percent confidence limits") #another example
error.bars(attitude,bar=TRUE)  #show the use of bar graphs


#combine with a strip chart and boxplot
stripchart(attitude,vertical=TRUE,method="jitter",jitter=.1,pch=19,main="Stripchart with
boxplot(attitude,add=TRUE)
error.bars(attitude,add=TRUE,arrow.len=.2)

#use statistics from somewhere else
my.stats <- data.frame(values=c(1,4,8),means=c(10,12,18),se=c(2,3,5))
error.bars(stats=my.stats,type="b",main="data with confidence intervals")
#note that in this case, the error bars are 1 s.e.  To modify that, change the s.e.

#Consider the case where we get stats from describe
temp <- describe(attitude)
error.bars(stats=temp)
#these error bars will be just one s.e.

#adjust the s.e. to vary by alpha level
alpha <- .05
```

```
temp[,"se"] <- temp[,"se"] * qt(1-alpha/2,temp[,"n"])
error.bars(stats=temp)
#show these do not differ from the other way by overlaying the two
error.bars(attitude,add=TRUE)
```

`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,y
```

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
`sd`	sd=TRUE will plot Standard Deviations instead of standard errors
`labels`	X axis label
`v.labels`	For a bar plot legend, these are the variable labels
`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
`bars`	Draw a barplot with error bars rather than a simple plot of the means
`within`	Should the s.e. be corrected by the correlation with the other variables?
`colors`	groups will be plotted in different colors (mod n.groups)
`lty`	line type may be specified in the case of not plotting by variables
`lines`	By default, when plotting different groups, connect the groups with a line of type = lty. If lines is FALSE, then do not connect the groups
`legend`	Where should the legend be drawn: 0 (do not draw it), 1= lower right corner, 2 = bottom, 3 ... 8 continue clockwise, 9 is the center
`...`	other parameters to pass to the plot function e.g., 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 was originally just a wrapper for `error.bars` but has been written to allow groups to be organized either as the x axis or as separate lines.

If desired, a barplot with error bars can be shown. Many find this type of plot to be uninformative (e.g., http://biostat.mc.vanderbilt.edu/DynamitePlots) and recommend the more standard dot plot.

Note in particular, if choosing to draw barplots, the starting value is 0.0 and setting the ylim parameter can lead to some awkward results if 0 is not included in the ylim range. Did you really mean to draw a bar plot in this case?

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.

These confidence regions are based upon normal theory and do not take into account any skew in the variables. More accurate confidence intervals could be found by resampling.

See Also

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

Examples

```
data(sat.act)
#The generic plot of variables by group
error.bars.by(sat.act[1:4],sat.act$gender,legend=7)
#a bar plot
error.bars.by(sat.act[5:6],sat.act$gender,bars=TRUE,labels=c("male","female"),main="SAT V
#a bar plot of SAT by age -- not recommended, see the next plot
error.bars.by(sat.act[5:6],sat.act$education,bars=TRUE,xlab="Education",main="95 percent
#a better graph uses points not bars
error.bars.by(sat.act[5:6],sat.act$education,TRUE, xlab="Education",legend=5,labels=colna

#now for a more complicated examples using 25 big 5 items scored into 5 scales
#and showing age trends by decade
#this shows how to convert many levels of a grouping variable (age) into more manageable
data(bfi)   #The Big 5 data
#first create the keys
 keys.list <- list(Agree=c(-1,2:5),Conscientious=c(6:8,-9,-10),Extraversion=c(-11,-12,13:
 keys <- make.keys(28,keys.list,item.labels=colnames(bfi))
 #then create the scores
 scores <- score.items(keys,bfi,min=1,max=6) #set the right limits for item reversals
 #now draw the results by age
 error.bars.by(scores$scores,round(bfi$age/10)*10,by.var=TRUE,main="BFI age trends",legen
```

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

Description

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

Usage

```
error.crosses(x,y,labels=NULL,main=NULL,xlim=NULL,ylim= NULL,xlab=NULL,ylab=NULL
```

Arguments

`x`	A vector of data or summary statistics (from Describe)
`y`	A second vector of data or summary statistics (also from Describe)
`labels`	the names of each pair – defaults to rownames of x
`main`	The title for the graph
`xlim`	xlim values if desired– defaults to min and max mean(x) +/- 2 se
`ylim`	ylim values if desired – defaults to min and max mean(y) +/- 2 se
`xlab`	label for x axis – grouping variable 1
`ylab`	label for y axis – grouping variable 2
`pos`	Labels are located where with respect to the mean?
`offset`	Labels are then offset from this location
`arrow.len`	Arrow length
`alpha`	alpha level of error bars
`sd`	if sd is TRUE, then draw means +/- 1 sd)
`...`	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

```
#just draw one pair of variables
desc <- describe(attitude)
x <- desc[1,]
y <- desc[2,]
error.crosses(x,y,xlab=rownames(x),ylab=rownames(y))

#now for a bit more complicated plotting
data(bfi)
desc <- describe.by(bfi[1:25],bfi$gender) #select a high and low group
error.crosses(desc$'1',desc$'2',ylab="female scores",xlab="male scores",main="BFI scores
 abline(a=0,b=1)

#do it from summary statistics  (using standard errors)
g1.stats <- data.frame(n=c(10,20,30),means=c(10,12,18),se=c(2,3,5))
g2.stats <- data.frame(n=c(15,20,25),means=c(6,14,15),se =c(1,2,3))
error.crosses(g1.stats,g2.stats)

#Or, if you prefer to draw +/- 1 sd. instead of 95% confidence
g1.stats <- data.frame(n=c(10,20,30),means=c(10,12,18),sd=c(2,3,5))
g2.stats <- data.frame(n=c(15,20,25),means=c(6,14,15),sd =c(1,2,3))
error.crosses(g1.stats,g2.stats,sd=TRUE)

#and seem even fancy plotting: This is taken from a study of mood
#four films were given (sad, horror, neutral, happy)
#with a pre and post test
data(affect)
colors <- c("black","red","white","blue")
films <- c("Sad","Horror","Neutral","Happy")
map.mat <- describe.by(maps[10:17],maps$Film,mat=TRUE)
 error.crosses(map.mat[c(1:4,17:20),],map.mat[c(5:8,21:24),],labels=films[map.mat$group1]
```

`fa` *MinRes (minimum residual) Factor analysis as well as Factor Analysis by Principal Axis, Weighted Least Squares or Maximum Likelihood*

Description

Among the many ways to do latent variable factor analysis, one of the better is to use Ordinary Least Squares to find the minimum residual (minres) solution. This produces solutions very similar to maximum likelihood even for badly behaved matrices. A variation on minres is to do weighted least squares. Perhaps the most conventional technique 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. Yet another estimate procedure is maximum likelihood. For well behaved matrices, maximum likelihood factor analysis (either in the fa or in the factanal fuction) is probably preferred. Bootstrapped confidence intervals the loadings and interfactor correlations are found by fa in n.iter > 1 and for dichotomous or polytomous items, by fa.poly.

Usage

```
fa(r,nfactors=1,n.obs = NA,n.iter=1, rotate="oblimin", scores=FALSE, residuals=F

fac(r,nfactors=1,n.obs = NA, rotate="oblimin", scores=FALSE, residuals=FALSE, SM

fa.poly(x,nfactors=1,n.obs = NA, n.iter=1, rotate="oblimin", SMC=TRUE,  missing=

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

factor.wls(r,nfactors=1,residuals=FALSE,rotate="varimax",n.obs = NA,
scores=FALSE,SMC=TRUE,missing=FALSE,impute="median", min.err = .001,digits=2,max
```

Arguments

r	A correlation matrix or a raw data matrix. If raw data, the correlation matrix will be found using pairwise deletion.
x	For fa.poly.ci, only raw data may be used
nfactors	Number of factors to extract, default is 1
n.obs	Number of observations used to find the correlation matrix if using a correlation matrix. Used for finding the goodness of fit statistics. Must be specified if using a correlaton matrix and finding confidence intervals.
rotate	"none", "varimax", "quartimax", "bentlerT", and "geominT" are orthogonal rotations. "promax", "oblimin", "simplimax", "bentlerQ, and "geominQ" or "cluster" are possible rotations or transformations of the solution. The default is to do a oblimin transformation, although prior versions defaulted to varimax.
n.iter	Number of bootstrap interations to do in fa or fa.poly
residuals	Should the residual matrix be shown
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.
covar	if covar is TRUE, factor the covariance matrix, otherwise factor the correlation matrix
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– deprecated – now specified in the print function
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
fm	factoring method fm="minres" will do a minimum residual (OLS), fm="wls" will do a weighted least squares (WLS) solution, fm="gls" does a generalized weighted least squares (GLS), fm="pa" will do the principal factor solution, fm="ml" will do a maximum likelihood factor analysis

alpha	alpha level for the confidence intervals for RMSEA
p	if doing iterations to find confidence intervals, what probability values should be found for the confidence intervals
oblique.scores	If factor scores are found, should they be based on the structure matrix (default) or the pattern matrix (which is what factanal seems to do).
...	additional parameters, specifically, keys may be passed if using the target rotation, or delta if using geominQ, or whether to normalize if using Varimax

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). If variables are thought to represent a "true" or latent part then factor analysis provides an estimate of the correlations with the latent factor(s) representing the data. If variables are thought to be measured without error, then principal components provides the most parsimonious description of the data.

The fa function will do factor analyses using one of four different algorithms: minimum residual (minres), principal axes, weighted least squares, or maximum likelihood.

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. In either case, iterative techniques will tend to converge on a stable solution. 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.

The minimum residual (minres) solution is an unweighted least squares solution that takes a slightly different approach. It uses the optim function and adjusts the diagonal elements of the correlation matrix to mimimize the squared residual when the factor model is the eigen value decomposition of the reduced matrix. MINRES and PA will both work when ML will not, for they can be used when the matrix is singular. At least on a number of test cases, the MINRES solution is slightly more similar to the ML solution than is the PA solution. To a great extent, the minres and wls solutions follow ideas in the factanal function.

The weighted least squares (wls) solution weights the residual matrix by 1/ diagonal of the inverse of the correlation matrix. This has the effect of weighting items with low communalities more than those with high communalities.

The generalized least squares (gls) solution weights the residual matrix by the inverse of the correlation matrix. This has the effect of weighting those variables with low communalities even more than those with high communalities.

The maximum likelihood solution takes yet another approach and finds those communality values that minimize the chi square goodness of fit test. The fm="ml" option provides a maximum likelihood solution following the procedures used in `factanal` but does not provide all the extra features of that function.

Test cases comparing the output to SPSS suggest that the PA algorithm matches what SPSS calls uls, and that the wls solutions are equivalent in their fits. The wls and gls solutions have slightly larger eigen values, but slightly worse fits of the off diagonal residuals than do the minres or maximum likelihood solutions.

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. There are multiple approaches that are possible (see Grice, 2001) and the one taken here is Thurstone's least squares regression where the weights are found by $W = R^(-1)S$ where R is the correlation matrix of the variables ans S is the structure matrix. Then, factor scores are just $Fs = XW$.

In the oblique case, the factor loadings are referred to as Pattern coefficients and are related to the Structure coefficients by $S = P\Phi$ and thus $P = S\Phi^{-1}$. When estimating factor scores, `fa` and `factanal` differ in that `fa` finds the factors from the Structure matrix and `factanal` seems to find them from the Pattern matrix. Thus, although in the orthogonal case, fa and factanal agree perfectly in their factor score estimates, they do not agree in the case of oblique factors. Setting oblique.scores = FALSE will produce factor score estimate that match those of `factanal`.

It is sometimes useful to extend the factor solution to variables that were not factored. This may be done using `fa.extension`. Factor extension is typically done in the case where some variables were not appropriate to factor, but factor loadings on the original factors are still desired.

For dichotomous items or polytomous items, it is recommended to analyze the `tetrachoric` or `polychoric` correlations rather than the Pearson correlations. This is done automatically when using `irt.fa` or `fa.poly` functions. In the first case, the factor analysis results are reported in Item Response Theory (IRT) terms, although the original factor solution is returned in the results. In the later case, a typical factor loadings matrix is returned, but the tetrachoric/polychoric correlation matrix and item statitics are saved for reanalysis by `irt.fa`

Of the various rotation/transformation options, varimax, Varimax, quartimax, bentlerT and geominT do orthogonal rotations. Promax transforms obliquely with a target matix equal to the varimax solution. oblimin, quartimin, simplimax, bentlerQ, and geominQ are oblique transformations. Most of these are just calls to the GPArotation package. The "cluster" option does a targeted rotation to a structure defined by the cluster representation of a varimax solution. With the optional "keys" parameter, the "target" option will rotate to a target supplied as a keys matrix. (See `target.rot`.)

There are two varimax rotation functions. One, Varimax, in the GPArotation package does not by default apply Kaiser normalization. The other, varimax, in the stats package, does. It appears that the two rotation functions produce slightly different results even when normalization is set. For consistency with the other rotation functions, Varimax is probably preferred.

When factor analyzing items with dichotomous or polytomous responses, the `irt.fa` function provides an Item Response Theory representation of the factor output.

The function `fa` will repeat the analysis n.iter times on a bootstrapped sample of the data (if they exist) or of a simulated data set based upon the observed correlation matrix. The mean estimate and standard deviation of the estimate are returned and will print the original factor analysis as well as the alpha level confidence intervals for the estimated coefficients. The bootstrapped solutions are rotated towards the original solution using target.rot. The factor loadings are z-transformed, averaged and then back transformed.

`fa.poly` will find confidence intervals for a factor solution for dichotomous or polytomous items (set n.iter > 1 to do so). Perhaps more useful is to find the Item Response Theory parameters equivalent to the factor loadings reported in fa.poly by using the `irt.fa` function.

Value

values	Eigen values of the common factor solution
e.values	Eigen values of the original matrix
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. To show these by sorted order, use print.psych with sort=TRUE
fit	How well does the factor model reproduce the correlation matrix. This is just $\frac{\Sigma r_{ij}^2 - \Sigma r_{ij}^{*2}}{\Sigma r_{ij}^2}$ (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 (For principal axis only.) 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.
rms	This is the sum of the squared (off diagonal residuals) divided by the degrees of freedom. Comparable to an RMSEA which, because it is based upon χ^2, requires the number of observations to be specified. The rms is an empirical value while the RMSEA is based upon normal theory and the non-central χ^2 distribution.
TLI	The Tucker Lewis Index of factoring reliability which is also known as the non-normed fit index.
BIC	Based upon χ^2
R2	The multiple R square between the factors and factor score estimates, if they were to be found. (From Grice, 2001). Derived from R2 is is the minimum correlation between any two factor estimates = 2R2-1.
r.scores	The correlations of the factor score estimates, if they were to be found.

weights	The beta weights to find the factor score estimates
valid	The validity coffiecient of course coded (unit weighted) factor score estimates (From Grice, 2001)
score.cor	The correlation matrix of course coded (unit weighted) factor score estimates, if they were to be found, based upon the loadings matrix rather than the weights matrix.

Note

Thanks to Erich Studerus for some very helpful suggestions about various rotation and factor scoring algorithms, and to Gumundur Arnkelsson for suggestions about factor scores for singular matrices.

The fac function is the original fa function which is now called by fa repeatedly to get confidence intervals.

Author(s)

William Revelle

References

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

Grice, James W. (2001), Computing and evaluating factor scores. Psychological Methods, 6, 430-450

Harman, Harry and Jones, Wayne (1966) Factor analysis by minimizing residuals (minres), Psychometrika, 31, 3, 351-368.

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

See Also

principal, irt.fa, VSS, ICLUST, predict.psych, fa.extension

Examples

```
#using the Harman 24 mental tests, compare a principal factor with a principal components
pc <- principal(Harman74.cor$cov,4,rotate="varimax")
pa <- fa(Harman74.cor$cov,4,fm="pa" ,rotate="varimax")  #principal axis
uls <- fa(Harman74.cor$cov,4,rotate="varimax")          #unweighted least squares is minr
wls <- fa(Harman74.cor$cov,4,fm="wls")       #weighted least squares

#to show the loadings sorted by absolute value
print(uls,sort=TRUE)

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

#finally, compare the unrotated factor, ml, uls, and  wls solutions
wls <- factor.wls(Harman74.cor$cov,4,rotate="none")
pa <- factor.pa(Harman74.cor$cov,4,rotate="none")
mle <-  factanal(factors=4,covmat=Harman74.cor$cov,rotation="none")
```

```
uls <- factor.minres(Harman74.cor$cov,4,rotate="none")

factor.congruence(list(mle,pa,uls,wls))
#note that the order of factors and the sign of some of factors differ



#an example of where the ML and PA and MR models differ is found in Thurstone.33.
#compare the first two factors with the 3 factor solution
Thurstone.33 <- as.matrix(Thurstone.33)
mle2 <- factanal(covmat=Thurstone.33,factors=2,rotation="none")
mle3 <- factanal(covmat=Thurstone.33,factors=3 ,rotation="none")
pa2 <- factor.pa(Thurstone.33,2,rotate="none")
pa3 <- factor.pa(Thurstone.33,3,rotate="none")
mr2 <- fa(Thurstone.33,2,rotate="none")
mr3 <- fa(Thurstone.33,3,rotate="none")
factor.congruence(list(mle2,mle3,pa2,pa3,mr2,mr3))
```

`fa.diagram` *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.diagram(fa.results,Phi=NULL,fe.results=NULL,sort=TRUE,labels=NULL,cut=.3,simp
    digits=1,e.size=.05,rsize=.15,side=2,main,cex=NULL, ...)
fa.graph(fa.results,out.file=NULL,labels=NULL,cut=.3,simple=TRUE,
   size=c(8,6), node.font=c("Helvetica", 14),
    edge.font=c("Helvetica", 10), rank.direction=c("RL","TB","LR","BT"), digits=
```

Arguments

`fa.results`	The output of factor analysis, principal components analysis, or ICLUST analysis. May also be a factor loading matrix from anywhere.
`Phi`	Normally not specified (it is is found in the FA, pc, or ICLUST, solution), this may be given if the input is a loadings matrix.
`fe.results`	the results of a factor extension analysis (if any)
`out.file`	If it exists, a dot representation of the graph will be stored here (fa.graph)
`labels`	Variable labels
`cut`	Loadings with abs(loading) > cut will be shown
`simple`	Only the biggest loading per item is shown
`size`	graph size
`sort`	sort the factor loadings before showing the diagram
`errors`	include error estimates (as arrows)

e.size	size of ellipses
rsize	size of rectangles
side	on which side should error arrows go?
cex	modify font size
node.font	what font should be used for nodes in fa.graph
edge.font	what font should be used for edges in fa.graph
rank.direction	parameter passed to Rgraphviz– which way to draw the graph
digits	Number of digits to show as an edgelable
main	Graphic title, defaults to "factor analyis" or "factor analysis and extension"
graphviz	Should we try to use Rgraphviz for output?
...	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.

fa.diagram does not use Rgraphviz and is the preferred function.

In fa.graph, 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 or done using fa.diagram.

The normal input is taken from the output of either fa or ICLUST. It is also possible to just give a factor loading matrix as input. In this case, supplying a Phi matrix of factor correlations is also possible.

To specify the model for a structural equation confirmatory analysis of the results, use structure.diagram instead.

Value

fa.diagram: A path diagram is drawn without using Rgraphviz. This is probably the more useful function.

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

Note

fa.graph requires Rgraphviz. Because there are occasional difficulties installing Rgraphviz from Bioconductor in that some libraries are misplaced and need to be relinked, it is probably better to use fa.diagram.

Author(s)

William Revelle

See Also

omega.graph, ICLUST.graph, structure.diagram to convert the factor diagram to sem modeling code.

Examples

```
test.simple <- fa(item.sim(16),2,rotate="oblimin")
#if(require(Rgraphviz)) {fa.graph(test.simple) }
fa.diagram(test.simple)
f3 <- fa(Thurstone,3,rotate="cluster")
fa.diagram(f3,cut=.4,digits=2)
f3l <- f3$loadings
fa.diagram(f3l,main="input from a matrix")
Phi <- f3$Phi
fa.diagram(f3l,Phi=Phi,main="Input from a matrix")
fa.diagram(ICLUST(Thurstone,2,title="Two cluster solution of Thurstone"),main="Input from
```

`fa.extension` *Apply Dwyer's factor extension to find factor loadings for extended variables*

Description

Dwyer (1937) introduced a method for finding factor loadings for variables not included in the original analysis. This is basically finding the unattenuated correlation of the extension variables with the factor scores. An alternative, which does not correct for factor reliability was proposed by Gorsuch (1997). Both options are an application of exploratory factor analysis with extensions to new variables.

Usage

```
fa.extension(Roe,fo,correct=TRUE)
```

Arguments

`Roe`	The correlations of the original variables with the extended variables
`fo`	The output from the `fa` or `omega` functions applied to the original variables.
`correct`	correct=TRUE produces Dwyer's solution, correct=FALSE produces Gorsuch's solution

Details

It is sometimes the case that factors are derived from a set of variables (the Fo factor loadings) and we want to see what the loadings of an extended set of variables (Fe) would be. Given the original correlation matrix Ro and the correlation of these original variables with the extension variables of Roe, it is a straight forward calculation to find the loadings Fe of the extended variables on the original factors. This technique was developed by Dwyer (1937) for the case of adding new variables to a factor analysis without doing all the work over again. But, as discussed by Horn (1973) factor extension is also appropriate when one does not want to include the extension variables in the original factor analysis, but does want to see what the loadings would be anyway.

This could be done by estimating the factor scores and then finding the covariances of the extension variables with the factor scores. But if the original data are not available, but just the covariance or correlation matrix is, then the use of `fa.extension` is most appropriate.

The factor analysis results from either fa or omega functions applied to the original correlation matrix is extended to the extended variables given the correlations (Roe) of the extended variables with the original variables.

fa.extension assumes that the original factor solution was found by the fa function.

For a very nice discussion of the relationship between factor scores, correlation matrices, and the factor loadings in a factor extension, see Horn (1973).

Value

Factor Loadings of the exended variables on the original factors

Author(s)

William Revelle

References

Paul S. Dwyer (1937), The determination of the factor loadings of a given test from the known factor loadings of other tests. Psychometrika, 3, 173-178

Gorsuch, Richard L. (1997) New procedure for extension analysis in exploratory factor analysis, Educational and Psychological Measurement, 57, 725-740

Horn, John L. (1973) On extension analysis and ite relation to correlations between variables and factor scores. Multivariate Behavioral Research, 8, (4), 477-489.

See Also

See Also as fa, principal, Dwyer

Examples

```
 #The Dwyer Example
Ro <- Dwyer[1:7,1:7]
Roe <- Dwyer[1:7,8]
fo <- fa(Ro,2,rotate="none")
fe <- fa.extension(Roe,fo)

#an example from simulated data
set.seed(42)
 d <- sim.item(12)    #two orthogonal factors
 R <- cor(d)
 Ro <- R[c(1,2,4,5,7,8,10,11),c(1,2,4,5,7,8,10,11)]
 Roe <- R[c(1,2,4,5,7,8,10,11),c(3,6,9,12)]
 fo <- fa(Ro,2)
 fe <- fa.extension(Roe,fo)
 fa.diagram(fo,fe=fe)

 #create two correlated factors
 fx <- matrix(c(.9,.8,.7,.85,.75,.65,rep(0,12),.9,.8,.7,.85,.75,.65),ncol=2)
 Phi <- matrix(c(1,.6,.6,1),2)
 sim.data <- sim.structure(fx,Phi,n=1000,raw=TRUE)
 R <- cor(sim.data$observed)
 Ro <- R[c(1,2,4,5,7,8,10,11),c(1,2,4,5,7,8,10,11)]
 Roe <- R[c(1,2,4,5,7,8,10,11),c(3,6,9,12)]
 fo <- fa(Ro,2)
```

```
fe <- fa.extension(Roe,fo)
fa.diagram(fo,fe=fe)

#an example of extending an omega analysis


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
gload <- matrix(rep(.7,5))
five.factor <- sim.hierarchical(gload,fload,500,TRUE) #create sample data set
ss <- c(1,2,3,5,6,7,9,10,11,13,14,15,17,18,19)
Ro <- cor(five.factor$observed[,ss])
Re <- cor(five.factor$observed[,ss],five.factor$observed[,-ss])
om5 <-omega(Ro,5)   #the omega analysis
fa.extension(Re,om5) #the extension analysis
```

`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 factors of the observed data with that of a random data matrix of the same size as the original. fa.parallel.poly does this for tetrachoric or polychoric analyses.

Usage

```
fa.parallel(x, n.obs = NULL,fm="minres", fa="both", main = "Parallel Analysis Sc
fa.parallel.poly(x ,n.iter=10,SMC=TRUE,  fm = "minres")
plot.poly.parallel(x,show.legend=TRUE,...)
```

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.
`fm`	What factor method to use. (minres, ml, uls, wls, gls, pa) See `fa` for details.
`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
`n.iter`	Number of simulated analyses to perform
`error.bars`	Should error.bars be plotted (default = FALSE)
`SMC`	SMC=TRUE finds eigen values after estimating communalities by using SMCs. smc = FALSE finds eigen values after estimating communalities with the first factor.

ylabel	Label for the y axis – defaults to “eigen values of factors and components", can be made empty to show many graphs
show.legend	the default is to have a legend. For multiple panel graphs, it is better to not show the legend
...	additional plotting parameters, for plot.poly.parallel

Details

Cattell’s “scree" test is one of most simple tests for the number of factors problem. Horn’s (1965) “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 the factor solution (minres by default) 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.

fa.parallel.poly will do parallel analysis for polychoric and tetrachoric factors. If the data are dichotomous, fa.parallel.poly will find tetrachoric correlations for the real and simulated data, otherwise, if the number of categories is less than 10, it will find polychoric correlations. Note that fa.parallel.poly is much slower than fa.parallel because of the complexity of calculating the tetrachoric/polychoric correlations.

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

Alternative ways to estimate the number of factors problem are discussed in the Very Simple Structure (Revelle and Rocklin, 1979) documentation (VSS) and include Wayne Velicer’s MAP algorithm (Veicer, 1976).

Parallel analysis for factors is actually harder than it seems, for the question is what are the appropriate communalities to use. If communalities are estimated by the Squared Multiple Correlation (SMC) smc, then the eigen values of the original data will reflect major as well as minor factors (see sim.minor to simulate such data). Random data will not, of course, have any structure and thus the number of factors will tend to be biased upwards by the presence of the minor factors.

By default, fa.parallel estimates the communalities based upon a one factor minres solution. Although this will underestimate the communalities, it does seem to lead to better solutions on simulated or real (e.g., the bfi or Harman74) data sets.

For comparability with other algorithms (e.g, the paran function in the paran package), setting smc=TRUE will use smcs as estimates of communalities. This will tend towards identifying more factors than the default option.

Printing the results will show the eigen values of the original data that are greater than simulated values.

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.

Also returned (invisibly) are:

fa.values	The eigen values of the factor model for the real data.
fa.sim	The descriptive statistics of the simulated factor models.
pc.values	The eigen values of a principal components of the real data.

pc.sim	The descriptive statistics of the simulated principal components analysis.
nfact	Number of factors with eigen values > eigen values of random data
ncomp	Number of components with eigen values > eigen values of random data

Author(s)

William Revelle

References

Floyd, Frank J. and Widaman, Keith. F (1995) Factor analysis in the development and refinement of clinical assessment instruments. Psychological Assessment, 7(3):286-299, 1995.

Horn, John (1965) A rationale and test for the number of factors in factor analysis. Psychometrika, 30, 179-185.

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.

Revelle, William and Rocklin, Tom (1979) Very simple structure - alternative procedure for estimating the optimal number of interpretable factors. Multivariate Behavioral Research, 14(4):403-414.

Velicer, Wayne. (1976) Determining the number of components from the matrix of partial correlations. Psychometrika, 41(3):321-327, 1976.

See Also

fa, VSS, VSS.plot, VSS.parallel, sim.minor

Examples

```
test.data <- Harman74.cor$cov
fa.parallel(test.data,n.obs=145)
set.seed(123)
minor <- sim.minor(24,4,400) #4 large and 12 minor factors
fa.parallel(minor$observed) #shows 4 factors  -- compare with
fa.parallel(minor$observed,SMC=TRUE) #which shows 8 factors
```

---

fa.sort *Sort factor analysis or principal components analysis loadings*

Description

Although the print.psych function will sort factor analysis loadings, sometimes it is useful to do this outside of the print function. fa.sort takes the output from the fa or principal functions and sorts the loadings for each factor. Items are located in terms of their greatest loading.

Usage

```
fa.sort(fa.results,polar=FALSE)
```

Arguments

fa.results	The output from a factor analysis or principal components analysis using fa or principal.
polar	Sort by polar coordinates of first two factors (FALSE)

Details

The fa.results$loadings are replaced with sorted loadings.

Value

A sorted factor analysis, principal components analysis, or omega loadings matrix.

These sorted values are used internally by the various diagram functions.

Author(s)

William Revelle

See Also

See Also as fa,print.psych, fa.diagram,

Examples

```
test.simple <- fa(item.sim(16),2,rotate="oblimin")
fa.sort(test.simple)
```

---

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=NULL,digits=2)
```

Arguments

x	A matrix of factor loadings or a list of matrices of factor loadings
y	A second matrix of factor loadings (if x is a list, then y may be empty)
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 of 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.

To compare more than two solutions, x may be a list of matrices, all of which will be compared.

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,Phi=NULL,U2=TRUE)
```

Arguments

f	A matrix of loadings.
Phi	A matrix of factor correlations
U2	Should the diagonal be model by ff' (U2 = TRUE) or replaced with 1's (U2 = FALSE)

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)
```

---

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,plot=FALSE,...)
```

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
plot	plot the original (unrotated) and rotated factors
...	parameters to pass to factor.plot

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. If the plot option is set to TRUE, then the original axes are shown as dashed lines.

The rotation is in degrees counter clockwise.

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/r/book

Examples

```
#using the Harman 24 mental tests, rotate the 2nd and 3rd factors 45 degrees
f4<- fa(Harman74.cor$cov,4,rotate="TRUE")
f4r45 <- factor.rotate(f4,45,2,3)
f4r90 <- factor.rotate(f4r45,45,2,3)
print(factor.congruence(f4,f4r45),digits=3) #poor congruence with original
print(factor.congruence(f4,f4r90),digits=3) #factor 2 and 3 have been exchanged and 3 fli

#a graphic example
data(Harman23.cor)
f2 <- fa(Harman23.cor$cov,2,rotate="none")
op <- par(mfrow=c(1,2))
cluster.plot(f2,xlim=c(-1,1),ylim=c(-1,1),title="Unrotated ")
f2r <- factor.rotate(f2,-33,plot=TRUE,xlim=c(-1,1),ylim=c(-1,1),title="rotated -33 degree
op <- par(mfrow=c(1,1))
```

`factor.stats` *Find various goodness of fit statistics for factor analysis and principal components*

Description

Chi square and other goodness of fit statistics are found based upon the fit of a factor or components model to a correlation matrix. Although these statistics are normally associated with a maximum likelihood solution, they can be found for minimal residual (OLS), principal axis, or principal component solutions as well. Primarily called from within these functions, factor.stats can be used by itself. Measures of factorial adequacy and validity follow the paper by Grice, 2001.

Usage

```
factor.stats(r, f, phi=NULL, n.obs = NA,alpha=.1)
factor.scores(x,f)
```

Arguments

`r`	A correlation matrix or a data frame of raw data
`x`	A data frame or matrix of raw data
`f`	A factor analysis loadings matrix or the output from a factor or principal components analysis
`phi`	A factor intercorrelation matrix if the factor solution was oblique
`n.obs`	The number of observations for the correlation matrix. If not specified, and a correlation matrix is used, chi square will not be reported. Not needed if the input is a data matrix.
`alpha`	alpha level of confidence intervals for RMSEA

Details

Combines the goodness of fit tests used in factor.pa, factor.minres, and principal into one function. If the matrix is singular, will smooth the correlation matrix before finding the fit functions. Now will find the RMSEA (root mean square error of approximation) and the alpha confidence intervals similar to a SEM function. Also reports the root mean square residual.

Value

`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.

`R2` The multiple R square between the factors and factor score estimates, if they were to be found. (From Grice, 2001)

`r.scores` The correlations of the factor score estimates, if they were to be found.

`weights` The beta weights to find the factor score estimates

`valid` The validity coffiecient of course coded (unit weighted) factor score estimates (From Grice, 2001)

`score.cor` The correlation matrix of course coded (unit weighted) factor score estimates, if they were to be found, based upon the loadings matrix.

`RMSEA` The Root Mean Square Error of Approximation and the alpha confidence intervals. Based upon the chi square non-centrality parameter.

`rms` The empirically found square root of the squared residuals. This does not require sample size to be specified nor does it make assumptions about normality.

`crms` While the rms uses the number of correlations to find the average, the crms uses the number of degrees of freedom. Thus, there is a subtle penalty for having too complex a model.

Author(s)

William Revelle

References

Grice, James W.,2001, Computing and evaluating factor scores, Psychological Methods, 6,4, 430-450.

See Also

factor.pa for principal axis factor analysis, factor.minres for minimum residual factor analysis, and principal for principal components.

Examples

```
v9 <- sim.hierarchical()
f3 <- factor.minres(v9,3)
factor.stats(v9,f3,n.obs=500)
f3o <- factor.pa(v9,3,rotate="Promax")
factor.stats(v9,f3o,n.obs=500)
```

---

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 <- 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 Consequences. 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.

See Also

The other Galton data sets: `heights`, `peas`,`cubits`

Examples

```
data(galton)
describe(galton)
pairs.panels(galton,main="Galton's Parent child heights")   #show the scatter plot and th
pairs.panels(galton,lm=TRUE,main="Galton's Parent child heights") #but this makes the req
pairs.panels(galton,lm=TRUE,xlim=c(62,74),ylim=c(62,74),main="Galton's Parent child heigh
```

`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,na.rm=TRUE)
```

Arguments

`x`	a vector or data.frame
`na.rm`	remove NA values before processing

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
x2[2] <- NA
X <- data.frame(x,x2)
geometric.mean(x)
geometric.mean(x2)
geometric.mean(X)
geometric.mean(X,na.rm=FALSE)
```

`glb.algebraic` *Find the greatest lower bound to reliability.*

Description

The greatest lower bound solves the “educational testing problem". That is, what is the reliability of a test? (See `guttman` for a discussion of the problem). Although there are many estimates of a test reliability (Guttman, 1945) most underestimate the true reliability of a test.

For a given covariance matrix of items, C, the function finds the greatest lower bound to reliability of the total score using the csdp function from the Rcsdp package.

Usage

```
glb.algebraic(Cov, LoBounds = NULL, UpBounds = NULL)
```

Arguments

`Cov`	A p * p covariance matrix. Positive definiteness is not checked.
`LoBounds`	A vector $l = (l_1, \ldots, l_p)$ of length p with lower bounds to the diagonal elements x_i. The default l=(0, . . . , 0) does not imply any constraint, because positive semidefiniteness of the matrix $\tilde{C} + Diag(x)$ implies $0 \leq x_i$
`UpBounds`	A vector u =(u1, . . . , up) of length p with upper bounds to the diagonal elements xi. The default is u = v.

Details

If C is a p * p-covariance matrix, v = diag(C) its diagonal (i. e. the vector of variances $v_i = c_{ii}$), $\tilde{C} = C - Diag(v)$ is the covariance matrix with 0s substituted in the diagonal and x = the vector $x_1, \ldots, x_n$ the educational testing problem is (see e. g., Al-Homidan 2008)

$$\sum_{i=1}^{p} x_i \rightarrow \min$$

s.t.

$$\tilde{C} + Diag(x) \geq 0$$

(i.e. positive semidefinite) and $x_i \leq v_i, i = 1, \ldots, p$. This is the same as minimizing the trace of the symmetric matrix

$$\tilde{C} + diag(x) = \begin{pmatrix} x_1 & c_{12} & \ldots & c_{1p} \\ c_{12} & x_2 & \ldots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{1p} & c_{2p} & \ldots & x_p \end{pmatrix}$$

s. t. $\tilde{C} + Diag(x)$ is positive semidefinite and $x_i \leq v_i$.

The greatest lower bound to reliability is

$$\frac{\sum_{ij} \bar{c_{ij}} + \sum_i x_i}{\sum_{ij} c_{ij}}$$

Additionally, function glb.algebraic allows the user to change the upper bounds $x_i \leq v_i$ to $x_i \leq u_i$ and add lower bounds $l_i \leq x_i$.

The greatest lower bound to reliability is applicable for tests with non-homogeneous items. It gives a sharp lower bound to the reliability of the total test score.

Caution: Though glb.algebraic gives exact lower bounds for exact covariance matrices, the estimates from empirical matrices may be strongly biased upwards for small and medium sample sizes.

glb.algebraic is wrapper for a call to function csdp of package Rcsdp (see its documentation).

If Cov is the covariance matrix of subtests/items with known lower bounds, rel, to their reliabilities (e. g. Cronbachs α), LoBounds can be used to improve the lower bound to reliability by setting LoBounds <- rel*diag(Cov).

Changing UpBounds can be used to relax constraints $x_i \leq v_i$ or to fix x_i-values by setting LoBounds[i] < -z; UpBounds[i] <- z.

Value

`glb`	The algebraic greatest lower bound
`solution`	The vector x of the solution of the semidefinite program. These are the elements on the diagonal of C.
`status`	Status of the solution. See documentation of csdp in package Rcsdp. If status is 2 or greater or equal than 4, no glb and solution is returned. If status is not 0, a warning message is generated.
`Call`	The calling string

Author(s)

Andreas Moltner
Center of Excellence for Assessment in Medicine/Baden-Wurttemberg
University of Heidelberg

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

References

Al-Homidan S (2008). Semidefinite programming for the educational testing problem. Central European Journal of Operations Research, 16:239-249.

Bentler PM (1972) A lower-bound method for the dimension-free measurement of internal consistency. Soc Sci Res 1:343-357.

Fletcher R (1981) A nonlinear programming problem in statistics (educational testing). SIAM J Sci Stat Comput 2:257-267.

Shapiro A, ten Berge JMF (2000). The asymptotic bias of minimum trace factor analysis, with applications to the greatest lower bound to reliability. Psychometrika, 65:413-425.

ten Berge, Socan G (2004). The greatest bound to reliability of a test and the hypothesis of unidimensionality. Psychometrika, 69:613-625.

See Also

For an alternative estimate of the greatest lower bound, see glb.fa. For multiple estimates of reliablity, see guttman

Examples

```
Cv<-matrix(c(215, 64, 33, 22,
             64, 97, 57, 25,
             33, 57,103, 36,
             22, 25, 36, 77),ncol=4)

Cv                    # covariance matrix of a test with 4 subtests
Cr<-cov2cor(Cv)       # Correlation matrix of tests
if(require(Rcsdp)) {glb.algebraic(Cv)}     # glb of total score
if(require(Rcsdp)) {glb.algebraic(Cr) }    # glb of sum of standardized scores

w<-c(1,2,2,1)         # glb of weighted total score
# glb.algebraic(diag(w) %*% Cv %*% diag(w))
alphas <- c(0.8,0,0,0) # Internal consistency of first test is known
if(require(Rcsdp)) {glb.algebraic(Cv,LoBounds=alphas*diag(Cv))}

                      # Fix all diagonal elements to 1 but the first:
if(require(Rcsdp)) {lb<-glb.algebraic(Cr,LoBounds=c(0,1,1,1),UpBounds=c(1,1,1,1))
lb$solution[1]        # should be the same as the squared mult. corr.
smc(Cr)[1]

} else {print ('I am sorry, you need to have the package Rcsdp installed to use glb.algeb
```

Gorsuch — *Example data set from Gorsuch (1997) for an example factor extension.*

Description

Gorsuch (1997) suggests an alternative to the classic Dwyer (1937) factor extension technique. This data set is taken from that article. Useful for comparing link{fa.extension} with and without the correct=TRUE option.

Usage

```
data(Gorsuch)
```

Details

Gorsuc (1997) suggested an alternative model for factor extension. His method is appropriate for the case of repeated variables. This is handled in link{fa.extension} with correct=FALSE

Source

Richard L. Gorsuch (1997) New Procedure for Extension Analysis in Exploratory Factor Analysis. Educational and Psychological Measurement, 57, 725-740.

References

Dwyer, Paul S. (1937), The determination of the factor loadings of a given test from the known factor loadings of other tests. Psychometrika, 3, 173-178

Examples

```
data(Gorsuch)

Ro <- Gorsuch[1:6,1:6]
Roe <- Gorsuch[1:6,7:10]
fo <- fa(Ro,2,rotate="none")
fa.extension(Roe,fo,correct=FALSE)
```

---

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)
tenberge(r)
glb(r,key=NULL)
glb.fa(r,key=NULL)
```

Arguments

r	A correlation or covariance matrix or raw data matrix.
key	a vector of -1, 0, 1 to select or reverse key items

Details

Surprisingly, 105 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).

Guttman's first estimate λ_1 assumes that all the variance of an item is error:

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

This is a clear underestimate.

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 and 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

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

.

The smc is found from all the items. A modification to Guttman λ_6, λ_6* reported by the score.items function is to find the smc from the entire pool of items given, not just the items on the selected scale.

Guttman's λ_4 is the greatest split half reliability. This is found here 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 clusters. 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.)

These three procedures will produce keys vectors for assigning items to the two splits. The maximum split half reliability is found by taking the maximum of these three approaches. This is not elegant but is fast.

There are three greatest lower bound functions. One, glb finds the greatest split half reliability, λ_4. This considers the test as set of items and examines how best to partition the items into splits. The other two, glb.fa and glb.algebraic, are alternative ways of weighting the diagonal of the matrix.

glb.fa estimates the communalities of the variables from a factor model where the number of factors is the number with positive eigen values. Then reliability is found by

$$glb = 1 - \frac{\sum e_j^2}{V_x} = 1 - \frac{\sum(1 - h^2)}{V_x}$$

This estimate will differ slightly from that found by glb.algebraic, written by Andreas Moeltner which uses calls to csdp in the Rcsdp package. His algorithm, which more closely matches the description of the glb by Jackson and Woodhouse, seems to have a positive bias (i.e., will over estimate the reliability of some items; they are said to be = 1) for small sample sizes. More exploration of these two algorithms is underway.

Compared to glb.algebraic, glb.fa seems to have less (positive) bias for smallish sample sizes (n < 500) but larger for large (> 1000) sample sizes. This interacts with the number of variables so that equal bias sample size differs as a function of the number of variables. The differences are, however small. As samples sizes grow, glb.algebraic seems to converge on the population value while glb.fa has a positive bias.

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
glb	glb found from factor analysis
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

alpha, omega, ICLUST, glb.algebraic

Examples

```
data(attitude)
glb(attitude)
glb.fa(attitude)
if(require(Rcsdp)) {glb.algebraic(cor(attitude)) }
guttman(attitude)
```

Harman — *Two data sets from Harman (1967). 9 cognitive variables from Holzinger and 8 emotional variables from Burt*

Description

Two classic data sets reported by Harman (1967) are 9 psychological (cognitive) variables taken from Holzinger and 8 emotional variables taken from Burt. Both of these are used for tests and demonstrations of various factoring algortithms.

Usage

```
data(Harman)
```

Details

- Harman.Holzinger: 9 x 9 correlation matrix of ability tests, N = 696.
- Harman.Burt: a 8 x 8 correlation matrix of “emotional" items. N = 172

Harman.Holzinger. The nine psychological variables from Harman (1967, p 244) are taken from unpublished class notes of K.J. Holzinger with 696 participants. This is a subset of 12 tests with 4 factors. It is yet another nice example of a bifactor solution. Bentler (2007) uses this data set to discuss reliablity analysis. The data show a clear bifactor structure and are a nice example of the various estimates of reliability included in the `omega` function. Should not be confused with the `Holzinger` or `Holzinger.9` data sets in `bifactor`.

Harman.Burt. Eight “emotional" variables are taken from Harman (1967, p 164) who in turn adapted them from Burt (1939). They are said be from 172 normal children aged nine to twelve. As pointed out by Harman, this correlation matrix is singular and has squared multiple correlations > 1. Because of this problem, it is a nice test case for various factoring algorithms. (For instance, omega will issue warning messages for fm="minres" or fm="pa" but will fail for fm="ml".)

The Burt data set probably has a typo in the original correlation matrix. Changing the Sorrow-Tenderness correlation from .87 to .81 makes the correlation positive definite.

As pointed out by Jan DeLeeuw, the Burt data set is a subset of 8 variables from the original 11 reported by Burt in 1915. That matrix has the same problem. See `burt`.

Other example data sets that are useful demonstrations of factor analysis are the seven bifactor examples in `bifactor` and the 24 ability measures in `Harman74.cor`

Source

Harman (1967 p 164 and p 244.)

References

Harman, Harry Horace (1967), Modern factor analysis. Chicago, University of Chicago Press.

P.Bentler. Covariance structure models for maximal reliability of unit-weighted composites. In Handbook of latent variable and related models, pages 1–17. North Holland, 2007.

Burt, C.General and Specific Factors underlying the Primary Emotions. Reports of the British Association for the Advancement of Science, 85th meeting, held in Manchester, September 7-11, 1915. London, John Murray, 1916, p. 694-696 (retrieved from the web at http://www.biodiversitylibrary.org/item/95822#790)

See Also

See also the original `burt` data set

Examples

```
data(Harman)
cor.plot(Harman.Holzinger)
cor.plot(Harman.Burt)
smc(Harman.Burt)  #note how this produces impossible results
```

`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,na.rm=TRUE)
```

Arguments

`x`	a vector, matrix, or data.frame
`na.rm`	na.rm=TRUE remove NA values before processing

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
x2[2] <- NA
X <- data.frame(x,x2)
harmonic.mean(x)
harmonic.mean(x2)
harmonic.mean(X)
harmonic.mean(X,FALSE)
```

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 text 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,missing=TRUE,alpha=.05)
```

Arguments

`x`	a matrix or dataframe of ratings
`missing`	if TRUE, remove missing data – work on complete cases only
`alpha`	The alpha level for significance for finding the confidence intervals

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 fixed effects model and is found by (MSB- MSW)/(MSB+ (nr-1)*MSW))

ICC2: A random sample of k judges rate each target. The measure is one of absolute agreement in the ratings. Found as (MSB- MSE)/(MSB + (nr-1)*MSE + nr*(MSJ-MSE)/nc)

ICC3: A fixed set of k judges rate each target. There is no generalization to a larger population of judges. (MSB - MSE)/(MSB+ (nr-1)*MSE)

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.

The intraclass correlation is used if raters are all of the same "class". That is, there is no logical way of distinguishing them. Examples include correlations between pairs of twins, correlations between raters. If the variables are logically distinguishable (e.g., different items on a test), then the more typical coefficient is based upon the inter-class correlation (e.g., a Pearson r) and a statistic such as `alpha` or `omega` might be used.

Value

`results`	A matrix of 6 rows and 8 columns, including the ICCs, F test, p values, and confidence limits
`summary`	The anova summary table
`stats`	The anova statistics
`MSW`	Mean Square Within based upon the anova

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 have been using the formula in McGraw and Wong, but not the errata on p 390. They seem to have fixed it in more recent releases (15).

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` *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
n.iterations = 0,title="iclust",plot=TRUE,weighted=TRUE,cor.gen=TRUE,SMC=TRUE)
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
n.iterations = 0,title="ICLUST",plot=TRUE,weighted=TRUE,cor.gen=TRUE,SMC=TRUE)


#iclust(r.mat)    #use all defaults
#iclust(r.mat,nclusters =3)    #use all defaults and if possible stop at 3 clust
#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. If NULL, then the colnames are used. If FALSE, then labels are V1 .. Vn
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)
weighted	Weight the intercluster correlation by the size of the two clusters (TRUE) or do not weight them (FALSE)
cor.gen	When correlating clusters with subclusters, base the correlations on the general factor (default) or general + group (cor.gen=FALSE)
SMC	When estimating cluster-item correlations, use the smcs as the estimate of an item communality (SMC=TRUE) or use the maximum correlation (SMC=FALSE).

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.

The original ICLUST program was written in FORTRAN to run on CDC and IBM mainframes and was then modified to run in PC-DOS. The R version of iclust is a completely new version written 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.

In June, 2009, the option of weighted versus unweighted beta was introduced. Unweighted beta calculates beta based upon the correlation between two clusters, corrected for test length using the Spearman-Brown prophecy formala, while weighted beta finds the average interitem correlation between the items within two clusters and then finds beta from this. That is, for two clusters A and B of size N and M with between average correlation rb, weighted beta is (N+M)^2 rb/(Va +Vb + 2Cab). Raw (unweighted) beta is 2rab/(1+rab) where rab = Cab/sqrt(VaVb). Weighted beta seems a more appropriate estimate and is now the default. Unweighted beta is still available for consistency with prior versions.

Also modified in June, 2009 was the way of correcting for item overlap when calculating the cluster-subcluster correlations for the graphic output. This does not affect the final cluster solution, but does produce slightly different path values. In addition, there are two ways to solve for the cluster - subcluster correlation.

Given the covariance between two clusters, Cab with average rab = Cab/(N*M), and cluster variances Va and Vb with Va = N + N*(N-1)*ra then the correlation of cluster A with the combined cluster AB is either

a) ((N^2)ra + Cab)/sqrt(Vab*Va) (option cor.gen=TRUE) or b) (Va - N + Nra + Cab)/sqrt(Vab*Va) (option cor.gen=FALSE)

The default is to use cor.gen=TRUE.

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. The default is to produce graphics using the Rgraphviz package (from BioConductor). Because this package is sometimes hard to install, there is an alternative option to write the output out using the dot language. This will create a graphic suitable for any viewing program that uses 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 the default 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.)

An alternative solution is to ask for a somewhat more stringent set of criteria and require an increase in the size of beta for all clusters greater than 3 variables. This produces a 4 cluster solution.

It is also possible to use the original parameter settings, but ask for a 4 cluster solution.

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.

Note

iclust draws graphical displays with or without using Rgraphiviz. Rgraphviz produces slightly better graphics and will also export in the dot language for further massaging of the graphic. If Rgraphviz is not installed, ICLUST.graph will issue a warning message but continue. If, however, Rgraphviz is not properly installed, ICLUST.graph will think it is available, try to produce a graph, and fail (ungracefully). The solution to this problem is to specify plot=FALSE and the create graphs using the dot language. See the last example. With the introduction of the `diagram` functions, iclust now draws using iclust.diagram which is not as pretty, but more stable.

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
ic.out1 <- ICLUST(test.data,beta=3,beta.size=3)  #use more stringent criteria
print(ic.out1)
plot(ic.out)    #this shows the spatial representation
ic.no.graph <- ICLUST(test.data,plot=FALSE)
dot.graph <- ICLUST.graph(ic.no.graph,out.file="test.ICLUST.graph.dot")    #use a dot grap
```

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'

...

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.diagram *Draw an ICLUST hierarchical cluster structure diagram*

Description

Given a cluster structure determined by ICLUST, create a graphic structural diagram using graphic functions in the psych package To create dot code to describe the ICLUST output with more precision, use ICLUST.graph. If Rgraphviz has been successfully installed, the alternative is to use ICLUST.rgraph.

Usage

```
iclust.diagram(ic, labels = NULL, short = FALSE, digits = 2, cex = NULL, min.siz
```

Arguments

ic	Output from ICLUST
labels	labels for variables (if not specified as rownames in the ICLUST output
short	if short=TRUE, variable names are replaced with Vn
digits	Round the path coefficients to digits accuracy
cex	The standard graphic control parameter for font size modifications. This can be used to make the labels bigger or smaller than the default values.

min.size	Don't provide statistics for clusters less than min.size
e.size	size of the ellipses with the cluster statistics.
colors	postive and negative
main	The main graphic title

Details

iclust.diagram provides most of the power of ICLUST.rgraph without the difficulties involved in installing Rgraphviz. It is called automatically from ICLUST.

Value

Graphical output summarizing the hierarchical cluster structure. The graph is drawn using the diagram functions (e.g., dia.curve, dia.arrow, dia.rect, dia.ellipse) created as a work around to Rgraphviz.

Note

Suggestions for improving the graphic output are welcome.

Author(s)

William Revelle

References

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

See Also

ICLUST

Examples

```
v9 <- sim.hierarchical()
v9c <- ICLUST(v9)
test.data <- Harman74.cor$cov
ic.out <- ICLUST(test.data)
```

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 =
```

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
ic.out <- ICLUST(test.data)   #uses iclust.diagram instead
```

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

---

income *US family income from US census 2008*

Description

US census data on family income from 2008

Usage

```
data(income)
```

Format

A data frame with 44 observations on the following 4 variables.

`value` lower boundary of the income group

`count` Number of families within that income group

`mean` Mean of the category

`prop` proportion of families

Details

The distribution of income is a nice example of a log normal distribution. It is also an interesting example of the power of graphics. It is quite clear when graphing the data that income statistics are bunched to the nearest 5K. That is, there is a clear sawtooth pattern in the data.

The all.income set is interpolates intervening values for 100-150K, 150-200K and 200-250K

Source

US Census: Table HINC-06. Income Distribution to $250,000 or More for Households: 2008

http://www.census.gov/hhes/www/cpstables/032009/hhinc/new06_000.htm

Examples

```
data(income)
with(income[1:40,], plot(mean,prop, main="US family income for 2008",xlab="income", ylab=
with (income[1:40,], points(lowess(mean,prop,f=.3),typ="l"))
describe(income)


with(all.income, plot(mean,prop, main="US family income for 2008",xlab="income", ylab="Pr
with (all.income[1:50,], points(lowess(mean,prop,f=.25),typ="l"))
#curve(100000* dlnorm(x, 10.8, .8), x = c(0,250000),ylab="Proportion")
```

`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
```

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.

This data set may be used to demonstrate item response functions, `tetrachoric` correlations, or `irt.fa`.

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)
#convert them to true false
iq.scrub <- scrub(iqitems,isvalue=0)  #first get rid of the zero responses
iq.tf <-  score.multiple.choice(iq.keys,iq.scrub,score=FALSE) #convert to wrong (0) and c
describe(iq.tf)
```

`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. A better procedure is to use `irt.fa`.

Usage

```
irt.person.rasch(diff, items)
irt.0p(items)
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
`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. Much better functions are in the ltm and eRm packages. Similar analyses can be done using `irt.fa`

Author(s)

William Revelle

See Also

sim.irt, sim.rasch, logistic, irt.fa, tetrachoric, irt.item.diff.rasch

---

irt.fa — *Item Response Analysis by factor analysis of tetrachoric/polychoric correlations*

Description

Although factor analysis and Item Response Theory seem to be very different models of binary data, they can provide equivalent parameter estimates of item difficulty and item discrimination. Tetrachoric correlations of a data set of dichotomous items may be factor analysed using a minimum residual or maximum likelihood factor analysis and the result loadings transformed to item discrimination parameters. The tau parameter from the tetrachoric correlation combined with the item factor loading may be used to estimate item difficulties. Similar analyses can be done for discrete item responses using the polychoric correlation.

Usage

```
irt.fa(x,nfactors=1,correct=TRUE,...)
irt.select(x,y)
```

Arguments

x	A data matrix of dichotomous or discrete items, or the result of tetrachoric or polychoric
nfactors	Defaults to 1 factor
correct	If true, then correct the tetrachoric correlations for continuity. (See tetrachoric).
y	the subset of variables to pick from the rho and tau output of a previous irt.fa analysis to allow for further analysis.
...	Additional parameters to pass to the factor analysis function

Details

The tetrachoric correlation matrix of dichotomous items may be factored using a (e.g.) minimum residual factor analyis function fa and the resulting loadings, λ_i are transformed to discriminations by $\alpha = \frac{\lambda_i}{\sqrt{1-\lambda_i^2}}$.

The difficulty parameter, δ may be found from the τ parameter of the tetrachoric or polychoric function.

$\delta_i = \frac{\tau_i}{\sqrt{1-\lambda_i^2}}$

Similar analyses may be done with discrete item responses using polychoric correlations and distinct estimates of item difficulty (location) for each item response.

The results may be shown graphically using plot. For plotting there are three options: type = "ICC" will plot the item characteristic respone function. type = "IIC" will plot the item information function, and type= "test" will plot the test information function.

The normal input is just the raw data. If, however, the correlation matrix has already been found using tetrachoric or polychoric, then that result can be processed directly. Because irt.fa saves the rho and tau matrices from the analysis, subsequent analyses of the same data set are much faster if the input is the object returned on the first run. A similar feature is available in omega.

The output is best seen in terms of graphic displays. Plot the output from irt.fa to see item and test information functions.

The irt.select function is a helper function to allow for selecting a subset of a prior analysis for further analysis. First run irt.fa, then select a subset of variables to be analyzed in a subsequent irt.fa analysis. Perhaps a better approach is to just plot and find the information for selected items.

The plot function for an irt.fa object will plot ICC (item characteristic curves), IIC (item information curves), or test information curves. In addition, by using the "keys" option, these three kinds of plots can be done for selected items. This is particularly useful when trying to see the information characteristics of short forms of tests based upon the longer form factor analysis.

The plot function will also return (invisibly) the average information (area under the curve) for each item. These may be then printed or printed in sorted order using the sort option in print.

Value

irt	A list of Item location (difficulty) and discrimination
fa	A list of statistics for the factor analyis
rho	The tetrachoric/polychoric correlation matrix
tau	The tetrachoric/polychoric cut points

Note

Still under development. Comments welcome

In comparing irt.fa to the ltm function in the ltm package, the discrimination parameters are not identical, although the difficulty estimates are (although reversed). This is particularly noticeable in when analyzing the lsat6 data set.

Author(s)

William Revelle

References

McDonald, Roderick P. (1999) Test theory: A unified treatment. L. Erlbaum 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/

See Also

fa, sim.irt, tetrachoric, polychoric as well as plot.psych for plotting the IRT item curves.

Examples

```
set.seed(17)
d9 <- sim.irt(9,1000,-2.5,2.5,mod="normal") #dichotomous items
test <- irt.fa(d9$items)
```

```
test
op <- par(mfrow=c(3,1))
plot(test,type="ICC")
plot(test,type="IIC")
plot(test,type="test")
par(op)
set.seed(17)
items <- sim.congeneric(N=500,short=FALSE,categorical=TRUE) #500 responses to 4 discrete
d4 <- irt.fa(items$observed)  #item response analysis of congeneric measures


op <- par(mfrow=c(2,2))
plot(d4,type="ICC")
par(op)


#using the iq data set for an example of real items
#first need to convert the responses to tf
data(iqitems)
iq.keys <- c(4,4,3,1,4,3,2,3,1,4,1,3,4,3)
iq.tf <- score.multiple.choice(iq.keys,iqitems,score=FALSE)  #just the responses
iq.irt <- irt.fa(iq.tf)
plot(iq.irt)
#select a subset of these variables
small.iq.irt <- irt.select(iq.irt,c(1,5,9,10,11,13))
small.irt <- irt.fa(small.iq.irt)
plot(small.irt)
#find the information for three subset of iq items
keys <- make.keys(14,list(all=1:14,some=c(1,5,9,10,11,13),others=c(1:5)))
plot(iq.irt,keys=keys)
```

`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. A better procedure is to use `irt.fa` or the ltm package.

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.

The `irt.fa` function is also an alternative. This will find `tetrachoric` or `polychoric` correlations and then convert to IRT parameters using factor analysis (`fa`).

Value

a vector of item difficulties or item discriminations.

Note

Under development. Not recommended for public consumption. See `irt.fa` for a better option.

Author(s)

William Revelle

See Also

`irt.fa`, `irt.person.rasch`

---

logistic *Logistic transform from x to p and logit transform from p to x*

Description

The logistic function (1/(1+exp(-x)) and logit function (log(p/(1-p)) are fundamental to Item Response Theory. Although just one line functions, they are included here for ease of demonstrations and in drawing IRT models. Also included is the logistic.grm for a graded response model.

Usage

```
logistic(x,d=0, a=1,c=0, z=1)
logit(p)
logistic.grm( x,d=0,a=1.5,c=0,z=1,r=2,s=c(-1.5,-.5,.5,1.5))
```

Arguments

x	Any integer or real value
d	Item difficulty or delta parameter
a	The slope of the curve at x=0 is equivalent to the discrimination parameter in 2PL models or alpha parameter. Is either 1 in 1PL or 1.702 in 1PN approximations.
c	Lower asymptote = guessing parameter in 3PL models or gamma
z	The upper asymptote — in 4PL models
p	Probability to be converted to logit value
r	The response category for the graded response model
s	The response thresholds

Details

These three functions are provided as simple helper functions for demonstrations of Item Response Theory. The one parameter logistic (1PL) model is also known as the Rasch model. It assumes items differ only in difficulty. 1PL, 2PL, 3PL and 4PL curves may be drawn by choosing the appropriate d (delta or item difficulty), a (discrimination or slope), c (gamma or guessing) and z (zeta or upper asymptote).

logit is just the inverse of logistic.

logistic.grm will create the responses for a graded response model for the rth category where cutpoints are in s.

Value

p	logistic returns the probability associated with x
x	logit returns the real number associated with p

Author(s)

William Revelle

Examples

```
curve(logistic(x,a=1.702),-3,3,ylab="Probability of x",main="Logistic transform of x",xla
curve(pnorm(x),add=TRUE,lty="dashed")
curve(logistic(x),add=TRUE)
text(2,.8, expression(alpha ==1))
text(2,1.0,expression(alpha==1.7))
curve(logistic(x),-4,4,ylab="Probability of x", main = "Logistic transform of x in logit
curve(logistic(x,d=-1),add=TRUE)
curve(logistic(x,d=1),add=TRUE)
curve(logistic(x,c=.2),add=TRUE,lty="dashed")
text(1.3,.5,"d=1")
text(.3,.5,"d=0")
text(-1.5,.5,"d=-1")
text(-3,.3,"c=.2")
#demo of graded response model
 curve(logistic.grm(x,r=1),-4,4,ylim=c(0,1),main="Five level response scale",ylab="Probab
 curve(logistic.grm(x,r=2),add=TRUE)
 curve(logistic.grm(x,r=3),add=TRUE)
```

```
curve(logistic.grm(x,r=4),add=TRUE)
curve(logistic.grm(x,r=5),add=TRUE)

text(-2.,.5,1)
text(-1.,.4,2)
text(0,.4,3)
text(1.,.4,4)
 text(2.,.5,5)
```

`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 correlation matrix using `cluster.cor`, it is necessary to create a keys matrix. This is just a short cut for doing so. The keys matrix is a nvar x nscales matrix of -1,0, 1 and defines the membership for each scale.

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)
 scores

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

`mardia` *Calculate univariate or multivariate (Mardia's test) skew and 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. mardia applies Mardia's tests for multivariate skew and kurtosis

Usage

```
skew(x, na.rm = TRUE)
kurtosi(x, na.rm = TRUE,type=1)
mardia(x,na.rm = TRUE,plot=TRUE)
```

Arguments

`x`	A data.frame or matrix
`na.rm`	how to treat missing data
`type`	type=1 gives an unbiased estimate of kurtosis, type 2 gives a biased estimate.
`plot`	Plot the expected normal distribution values versus the Mahalanobis distance of the subjects.

Details

given a matrix or data.frame x, find the skew or kurtosis for each column (for skew and kurtosis) or the multivariate skew and kurtosis in the case of mardia.

As of revision 1.0.93, kurtosi by default gives an unbiased estimate of the kurtosis (DeCarlo, 1997). Prior versions used a different equation which produced a biased estimate. (See the kurtosis function in the e1071 package for the distinction between these two formulae. The default, type 1 is what is called type 2 in e1071. The other is their type 3.) For comparison with previous releases, specifying type = 2 will give the old estimate.

Value

skew	if input is a matrix or data.frame, skew is a vector of skews
kurtosi	if input is a matrix or data.frame, kurtosi is a vector of kurtosi
bp1	Mardia's bp1 estimate of multivariate skew
bp2	Mardia's bp2 estimate of multivariate kurtosis
skew	Mardia's skew statistic
small.skew	Mardia's small sample skew statistic
p.skew	Probability of skew
p.small	Probability of small.skew
kurtosis	Mardia's multivariate kurtosis statistic
p.kurtosis	Probability of kurtosis statistic
D	Mahalanobis distance of cases from centroid

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

References

L.DeCarlo. 1997) On the meaning and use of kurtosis, Psychological Methods, 2(3):292-307,

K.V. Mardia (1970). Measures of multivariate skewness and kurtosis with applications. Biometrika, 57(3):pp. 519-30, 1970.

See Also

describe, describe.by, mult.norm in QuantPsyc, Kurt in QuantPsyc

Examples

```
round(skew(attitude),2)
round(kurtosi(attitude),2)
mardia(attitude)
x <- matrix(rnorm(1000),ncol=10)
describe(x)
mardia(x)
```

mat.sort *Sort the elements of a correlation matrix to reflect factor loadings*

Description

To see the structure of a correlation matrix, it is helpful to organize the items so that the similar items are grouped together. One such grouping technique is factor analysis. mat.sort will sort the items by a factor model (if specified), or any other order, or by the loadings on the first factor (if unspecified)

Usage

```
mat.sort(m, f = NULL)
```

Arguments

m	A correlation matrix
f	A factor analysis output (i.e., one with a loadings matrix) or a matrix of weights

Details

The factor analysis output is sorted by size of the largest factor loading for each variable and then the matrix items are organized by those loadings. The default is to sort by the loadings on the first factor. Alternatives allow for ordering based upon any vector or matrix.

Value

A sorted correlation matrix, suitable for showing with cor.plot.

Author(s)

William Revelle

See Also

fa, cor.plot

Examples

```
data(bifactor)
sorted <- mat.sort(Bechtoldt.1,fa(Bechtoldt.1,5))
cor.plot(sorted)
```

`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="+")
```

`mixed.cor` *Find correlations for mixtures of continuous, polytomous, and dichotomous variables*

Description

For data sets with continuous, polytomous and dichotmous variables, the absolute Pearson correlation is downward biased from the underlying latent correlation. mixed.cor finds Pearson correlations for the continous variables, `polychoric`s for the polytomous items, `tetrachoric`s for the dichotomous items, and the `polyserial` or `biserial` correlations for the various mixed variables. Results include the complete correlation matrix, as well as the separate correlation matrices and difficulties for the polychoric and tetrachoric correlations.

Usage

```
mixed.cor(x = NULL, p = NULL, d = NULL)
```

Arguments

`x`	A set of continuous variables (may be missing)
`p`	A set of polytomous items (may be missing)
`d`	A set of dichotomous items (may be missing)

Details

This function is particularly useful as part of the Synthetic Apeture Personality Assessment (SAPA) data sets where continuous variables (age, SAT V, SAT Q, etc) and mixed with polytomous personality items taken from the International Personality Item Pool (IPIP) and the dichotomous experimental IQ items that have been developed as part of SAPA (see, e.g., Revelle, Wilt and Rosenthal, 2010).

Item response analyses using `irt.fa` may be done separately on the polytomous and dichotomous items in order to develop internally consistent scales. These scale may, in turn, be correlated with each other using the complete correlation matrix found by mixed.cor and using the `score.items` function.

This function is not quite as flexible as the hetcor function in John Fox's polychor package.

Note that the variables must be organized by type of data: first continuous, then polytomous, then dichotomous.

Value

`rho`	The complete matrix
`rx`	The Pearson correlation matrix for the continuous items
`poly`	the polychoric correlation (poly$rho) and the item difficulties (poly$tau)
`tetra`	the tetrachoric correlation (tetra$rho) and the item difficulties (tetra$tau)

Note

Note that the variables must be organized by type of data: first continuous, then polytomous, then dichotomous.

Author(s)

William Revelle

References

W.Revelle, J.Wilt, and A.Rosenthal. Personality and cognition: The personality-cognition link. In A.Gruszka, G. Matthews, and B. Szymura, editors, Handbook of Individual Differences in Cognition: Attention, Memory and Executive Control, chapter 2, pages 27-49. Springer, 2010.

See Also

polychoric, tetrachoric, score.items

Examples

```
data(bfi)
r <- mixed.cor(bfi[28],bfi[1:5],bfi[26])
round(r$rho,2)
#compare to raw Pearson
#note that the biserials and polychorics are not attenuated
rp <- cor(bfi[c(28,1:5,26)],use="pairwise")
round(rp,2)
```

---

msq *75 mood items from the Motivational State Questionnaire for 3896 participants*

Description

Emotions may be described either as discrete emotions or in dimensional terms. The Motivational State Questionnaire (MSQ) was developed to study emotions in laboratory and field settings. The data can be well described in terms of a two dimensional solution of energy vs tiredness and tension versus calmness. Additional items include what time of day the data were collected and a few personality questionnaire scores.

Usage

```
data(msq)
```

Format

A data frame with 3896 observations on the following 92 variables.

`MSQ_Time` Time of day the data were collected

`active` a numeric vector

`afraid` a numeric vector

`alert` a numeric vector

`alone` a numeric vector

`angry` a numeric vector

`aroused` a numeric vector

`ashamed` a numeric vector

`astonished` a numeric vector

`at-ease` a numeric vector

`at-rest` a numeric vector

`attentive` a numeric vector

`blue` a numeric vector

`bored` a numeric vector

`calm` a numeric vector

`clutched-up` a numeric vector

`confident` a numeric vector

`content` a numeric vector

`delighted` a numeric vector

`depressed` a numeric vector

`determined` a numeric vector

`distressed` a numeric vector

`drowsy` a numeric vector

`dull` a numeric vector

`elated` a numeric vector

`energetic` a numeric vector

`enthusiastic` a numeric vector

`excited` a numeric vector

`fearful` a numeric vector

`frustrated` a numeric vector

`full-of-pep` a numeric vector

`gloomy` a numeric vector

`grouchy` a numeric vector

`guilty` a numeric vector

`happy` a numeric vector

`hostile` a numeric vector

`inspired` a numeric vector

`intense` a numeric vector

`interested` a numeric vector

`irritable` a numeric vector

`jittery` a numeric vector

`kindly` a numeric vector

`lively` a numeric vector

`lonely` a numeric vector

`nervous` a numeric vector

`placid` a numeric vector

`pleased` a numeric vector

`proud` a numeric vector
`quiescent` a numeric vector
`quiet` a numeric vector
`relaxed` a numeric vector
`sad` a numeric vector
`satisfied` a numeric vector
`scared` a numeric vector
`scornful` a numeric vector
`serene` a numeric vector
`sleepy` a numeric vector
`sluggish` a numeric vector
`sociable` a numeric vector
`sorry` a numeric vector
`still` a numeric vector
`strong` a numeric vector
`surprised` a numeric vector
`tense` a numeric vector
`tired` a numeric vector
`unhappy` a numeric vector
`upset` a numeric vector
`vigorous` a numeric vector
`wakeful` a numeric vector
`warmhearted` a numeric vector
`wide-awake` a numeric vector
`anxious` a numeric vector
`idle` a numeric vector
`cheerful` a numeric vector
`inactive` a numeric vector
`tranquil` a numeric vector
`EA` Thayer's Energetic Arousal Scale
`TA` Thayer's Tense Arousal Scale
`PA` Positive Affect scale
`NegAff` Negative Affect scale
`Extraversion` Extraversion from the Eysenck Personality Inventory
`Neuroticism` Neuroticism from the Eysenck Personality Inventory
`Lie` Lie from the EPI
`Sociability` The sociability subset of the Extraversion Scale
`Impulsivity` The impulsivity subset of the Extraversions Scale
`MSQ_Round` Rounded time of day
`scale` a factor with levels `msq` `r` original or revised msq

`ID` subject ID

`exper` Which study were the data collected: a factor with levels `AGES` `BING` `BORN` `CART` `CITY` `COPE` `EMIT` `FAST` `Fern` `FILM` `FLAT` `Gray` `imps` `item` `knob` `MAPS` `mite` `pat-1` `pat-2` `PATS` `post` `RAFT` `Rim.1` `Rim.2` `rob-1` `rob-2` `ROG1` `ROG2` `SALT` `sam-1` `sam-2` `SAVE/PATS` `sett` `swam` `swam-2` `TIME` `VALE-1` `VALE-2` `VIEW`

`condition` What was the experimental condition after the msq was given

`TOD` a numeric vector

`TOD24` a numeric vector

Details

The Motivational States Questionnaire (MSQ) is composed of 72 items, which represent the full affective range (Revelle & Anderson, 1996). The MSQ consists of 20 items taken from the Activation-Deactivation Adjective Check List (Thayer, 1986), 18 from the Positive and Negative Affect Schedule (PANAS, Watson, Clark, & Tellegen, 1988) along with the items used by Larsen and Diener (1992). The response format was a four-point scale that corresponds to Russell and Carroll's (1999) "ambiguous–likely-unipolar format" and that asks the respondents to indicate their current standing ("at this moment") with the following rating scale:
0————————-1————————-2————————-3
Not at all A little Moderately Very much
The original version of the MSQ included 70 items. Intermediate analyses (done with 1840 subjects) demonstrated a concentration of items in some sections of the two dimensional space, and a paucity of items in others. To begin correcting this, 3 items from redundantly measured sections (alone, kindly, scornful) were removed, and 5 new ones (anxious, cheerful, idle, inactive, and tranquil) were added. Thus, the correlation matrix is missing the correlations between items 5, 42, and 55 and 72-76.

Procedure. The data were collected over nine years, as part of a series of studies examining the effects of personality and situational factors on motivational state and subsequent cognitive performance. In each of 38 studies, prior to any manipulation of motivational state, participants signed a consent form and filled out the MSQ. (The procedures of the individual studies are irrelevant to this data set and could not affect the responses to the MSQ, since this instrument was completed before any further instructions or tasks).

In addition to the MSQ, there are 5 scales from the Eysenck Personality Inventory.

Source

Data collecte at the Personality, Motivation, and Cognition Laboratory, Northwestern University.

References

William Revelle and Kristen Joan Anderson (1997) Personality, motivation and cognitive performance: Final report to the Army Research Institute on contract MDA 903-93-K-0008

Rafaeli, Eshkol and Revelle, William (2006), A premature consensus: Are happiness and sadness truly opposite affects? Motivation and Emotion, 30, 1, 1-12.

Examples

```
data(msq)
describe(msq)
```

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 Nor
```

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

See Also

bi.bars for drawing pairwise histograms

Examples

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

neo *NEO correlation matrix from the NEO_PI_R manual*

Description

The NEO.PI.R is a widely used personality test to assess 5 broad factors (Neuroticism, Extraversion, Openness, Agreeableness and Conscientiousness) with six facet scales for each factor. The correlation matrix of the facets is reported in the NEO.PI.R manual for 1000 subjects.

Usage

```
data(neo)
```

Format

A data frame of a 30 x 30 correlation matrix with the following 30 variables.

N1 Anxiety
N2 AngryHostility
N3 Depression
N4 Self-Consciousness
N5 Impulsiveness
N6 Vulnerability
E1 Warmth
E2 Gregariousness
E3 Assertiveness
E4 Activity
E5 Excitement-Seeking
E6 PositiveEmotions
O1 Fantasy
O2 Aesthetics
O3 Feelings
O4 Ideas
O5 Actions
O6 Values
A1 Trust
A2 Straightforwardness
A3 Altruism
A4 Compliance
A5 Modesty
A6 Tender-Mindedness
C1 Competence
C2 Order
C3 Dutifulness
C4 AchievementStriving
C5 Self-Discipline
C6 Deliberation

Details

The past thirty years of personality research has led to a general consensus on the identification of major dimensions of personality. Variously known as the “Big 5" or the “Five Factor Model", the general solution represents 5 broad domains of personal and interpersonal experience. Neuroticism and Extraversion are thought to reflect sensitivity to negative and positive cues from the environment and the tendency to withdraw or approach. Openness is sometimes labeled as Intellect and reflects an interest in new ideas and experiences. Agreeableness and Conscientiousness reflect tendencies to get along with others and to want to get ahead.

The factor structure of the NEO suggests five correlated factors as well as two higher level factors. The NEO was constructed with 6 “facets" for each of the five broad factors.

Source

Costa, Paul T. and McCrae, Robert R. (1992) (NEO PI-R) professional manual. Psychological Assessment Resources, Inc. Odessa, FL. (with permission of the author and the publisher)

References

Digman, John M. (1990) Personality structure: Emergence of the five-factor model. Annual Review of Psychology. 41, 417-440.

John M. Digman (1997) Higher-order factors of the Big Five. Journal of Personality and Social Psychology, 73, 1246-1256.

McCrae, Robert R. and Costa, Paul T., Jr. (1999) A Five-Factor theory of personality. In Pervin, Lawrence A. and John, Oliver P. (eds) Handbook of personality: Theory and research (2nd ed.) 139-153. Guilford Press, New York. N.Y.

Revelle, William (1995), Personality processes, Annual Review of Psychology, 46, 295-328.

Joshua Wilt and William Revelle (2009) Extraversion and Emotional Reactivity. In Mark Leary and Rick H. Hoyle (eds). Handbook of Individual Differences in Social Behavior. Guilford Press, New York, N.Y.

Examples

```
data(neo)
n5 <- factor.minres(neo,5)
neo.keys <- make.keys(30,list(N=c(1:6),E=c(7:12),O=c(13:18),A=c(19:24),C=c(25:30)))
n5p <- target.rot(n5,neo.keys) #show a targeted rotation for simple structure
n5p
```

---

omega — *Calculate McDonald's omega estimates of general and total 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). A related option is to define the model using omega and then perform a confirmatory factor analysis using the sem package. This is done by omegaSem and omegaFromSem.

Usage

```
omega(m,nfactors=3,fm="minres",n.iter=1,p=.05,poly=FALSE,key=NULL,flip=TRUE,digi
omegaSem(m,nfactors=3,fm="minres",key=NULL,flip=TRUE, digits=2,title="Omega",sl=
omegaFromSem(m,s,flip=TRUE)
omegah(m,nfactors=3,fm="minres",key=NULL,flip=TRUE, digits=2,title="Omega",sl=TR
```

Arguments

m	A correlation matrix, or a data.frame/matrix of data, or (if Phi is specified, an oblique factor pattern matrix
nfactors	Number of factors believed to be group factors
n.iter	How many replications to do in omega for bootstrapped estimates
fm	factor method (the default is minres) fm="pa" for principal axes, fm="minres" for a minimum residual (OLS) solution, fm="pc" for principal components, fm="ml" for maximum likelihood.
poly	should the correlation matrix be found using polychoric/tetrachoric or normal Pearson correlations
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.
p	probability of two tailed conference boundaries
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.diagram, plot =FALSE does not. If Rgraphviz is available, then omega.graph may be used separately.
n.obs	Number of observations - used for goodness of fit statistic
rotate	What rotation to apply? The default is oblimin, the alternatives include simplimax, Promax, cluster and target. target will rotate to an optional keys matrix (See target.rot)
Phi	If specified, then omega is found from the pattern matrix (m) and the factor intercorrelation matrix (Phi).
option	In the two factor case (not recommended), should the loadings be equal, emphasize the first factor, or emphasize the second factor. See in particular the option parameter in schmid for treating the case of two group factors.
...	Allows additional parameters to be passed through to the factor routines.
s	The output from a sem analysis

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. Four options are available, the default does a minres factor solution, fm="pa" does a principle axes factor analysis (factor.pa), fm="mle" uses the factanal function, and fm="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. Arbitrarily reversing items this way can overestimate the general factor. (See the example with a simulated circumplex).

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 part of the psych package. (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.

Diagnostic statistics that reflect the quality of the omega solution include a comparison of the relative size of the g factor eigen value to the other eigen values, the percent of the common variance for each item that is general factor variance (p2), the mean of p2, and the standard deviation of p2. Further diagnostics can be done by describing (describe) the $schmid$sl results.

Although omega_h is uniquely defined only for cases where 3 or more subfactors are extracted, it is sometimes desired to have a two factor solution. By default this is done by forcing the schmid extraction to treat the two subfactors as having equal loadings.

There are three possible options for this condition: setting the general factor loadings between the two lower order factors to be "equal" which will be the sqrt(oblique correlations between the factors) or to "first" or "second" in which case the general factor is equated with either the first or second group factor. A message is issued suggesting that the model is not really well defined. This solution discussed in Zinbarg et al., 2007. To do this in omega, add the option="first" or option="second" to the call.

Although obviously not meaningful for a 1 factor solution, it is of course possible to find the sum of the loadings on the first (and only) factor, square them, and compare them to the overall matrix variance. This is done, with appropriate complaints.

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.

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

Another estimate reported is the omega for an infinite length test with a structure similar to the observed test. This is found by

$$\omega_l imit = \frac{\vec{1}c\vec{c}'\vec{1}}{\vec{1}c\vec{c}'\vec{1} + \vec{1}\vec{A}\vec{A}'\vec{1}'}$$

The input to omega may be a correlation matrix or a raw data matrix, or a factor pattern matrix with the factor intercorrelations (Phi) matrix.

omega is an exploratory factor analysis function that uses a Schmid-Leiman transformation. omegaSem first calls omega and then takes the Schmid-Leiman solution, converts this to a confirmatory sem model and then calls the sem package to conduct a confirmatory model. ω_h is then calculated from

the CFA output. Although for well behaved problems, the efa and cfa solutions will be practically identical, the CFA solution will not always agree with the EFA solution. In particular, the estimated R^2 will sometimes exceed 1. (An example of this is the Harman 24 cognitive abilities problem.)

In addition, not all EFA solutions will produce workable CFA solutions. Model misspecifications will lead to very strange CFA estimates.

omegaFromSem takes the output from a sem model and uses it to find ω_h. The estimate of factor indeterminacy, found by the multiple R^2 of the variables with the factors, will not match that found by the EFA model. In particular, the estimated R^2 will sometimes exceed 1. (An example of this is the Harman 24 cognitive abilities problem.)

Value

`omega hierarchical`
: The ω_h coefficient

`omega.lim`
: The limit of ω_h as the test becomes infinitly large

`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$phi`
: 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

`various fit statistics`
: various fit statistics, see output

Note

Requires the GPArotation package.

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

If the factor solution leads to an exactly orthogonal solution (probably only for demonstration data sets), then use the rotate="Promax" option to get a solution.

omegaSem requires the sem package. omegaFromSem uses the output from the sem package.

omega may be run on raw data, a correlation matrix, a polychoric correlation matrix (found by e.g., polychoric), or the output of a previous omega run. This last case is particularly useful when working with categorical data using the poly=TRUE option. For in this case, most of the time is spent in finding the correlation matrix. The matrix is saved as part of the omega output and may be used as input for subsequent runs. A similar feature is found in irt.fa where the output of one analysis can be taken as the input to the subsequent analyses.

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, 74, 1, 145-154.

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
 if(!require(GPArotation)) {message("Omega requires GPA rotation" )} else {my.omega <- om
 print(my.omega,digits=2)}

#create 9 variables with a hierarchical structure
v9 <- sim.hierarchical()
#with correlations of
round(v9,2)
#find omega
v9.omega <- omega(v9,digits=2)
v9.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 matrice
omega(sim2,2)  #this shows that in fact there is no general factor
omega(sim2,2,option="first") #but, if we define one of the two group factors as a general
#apply omega to analyze 6 mental ability tests
data(ability.cov)   #has a covariance matrix
omega(ability.cov$cov)
```

```
## End(Not run)
```

`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 and omega.diagram draw both such structures. Graphs are drawn directly onto the graphics window or expressed in “dot" commands for conversion to graphics using implementations of Graphviz (if using omega.graph).

Using Graphviz allows the user to clean up the Rgraphviz output. However, if Graphviz and Rgraphviz are not available, use omega.diagram.

See the other structural diagramming functions, `fa.diagram` and `structure.diagram`.

In addition

Usage

```
omega.diagram(om.results,sl=TRUE,sort=TRUE,labels=NULL,cut=.2,gcut,simple=TRUE,e
omega.graph(om.results, out.file = NULL,  sl = TRUE, labels = NULL, size = c(8,
```

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
`cex`	control font size
`color.lines`	Use black for positive, red for negative
`title`	Figure title
`main`	main figure caption
`...`	Other options to pass into the graphics packages

e.size	the size to draw the ellipses for the factors. This is scaled by the number of variables.
cut	Minimum path coefficient to draw
gcut	Minimum general factor path to draw
simple	draw just one path per item
sort	sort the solution before making the diagram
side	on which side should errors be drawn?
errors	show the error estimates
rsize	size of the rectangles

Details

While omega.graph requires the Rgraphviz package, omega.diagram does not. codeomega 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

omega.graph 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

#
#example hierarchical structure from Jensen and Weng
if(require(GPArotation) ) {jen.omega <- omega(make.hierarchical())}
```

`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. Points may be given different colors depending upon some grouping variable.

Usage

```
pairs.panels(x, smooth = TRUE, scale = FALSE, density=TRUE,ellipses=TRUE,digits
```

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 '.').
cor	If plotting regressions, should correlations be reported?
jiggle	Should the points be jittered before plotting?
factor	factor for jittering (1-5)
hist.col	What color should the histogram on the diagonal be?
show.points	If FALSE, do not show the data points, just the data ellipses and smoothed functions
...	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 6-10. It is particularly useful for an initial overview of the data.

To show different groups with different colors, use a plot character (pch) between 21 and 25 and then set the background color to vary by group. (See the second example).

When plotting more than about 10 variables, it is useful to set the gap parameter to something less than 1 (e.g., 0). Alternatively, consider using cor.plot

Whe plotting more than about 100-200 cases, it is useful to use the show.points=FALSE option.

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 which is the base from which pairs.panels is derived, cor.plot to do a heat map of correlations, and scatter.hist to draw a single correlation plot with histograms and best fitted lines.

Examples

```
pairs.panels(attitude)   #see the graphics window
data(iris)
pairs.panels(iris[1:4],bg=c("red","yellow","blue")[iris$Species],pch=21,main="Fisher Iris

pairs.panels(iris[1:4],bg=c("red","yellow","blue")[iris$Species],pch=21,main="Fisher Iris

#demonstrate not showing the data points
data(sat.act)
pairs.panels(sat.act,show.points=FALSE)

#show many variables with 0 gap between scatterplots
data(bfi)
pairs.panels(bfi,show.points=FALSE,gap=0)
```

---

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)
```

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

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/ 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).

See Also

The other Galton data sets: `heights`, `galton`,`cubits`

Examples

```
data(peas)
pairs.panels(peas,lm=TRUE,xlim=c(14,22),ylim=c(14,22),main="Galton's Peas")
describe(peas)
pairs.panels(peas,main="Galton's Peas")
```

`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.

For a very useful discussion of various measures of association given a 2 x 2 table, and why one should probably prefer the Yule coefficient, see Warren (2008).

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

References

Warrens, Matthijs (2008), On Association Coefficients for 2x2 Tables and Properties That Do Not Depend on the Marginal Distributions. Psychometrika, 73, 777-789.

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)
```

---

phi.demo *A simple demonstration of the Pearson, phi, and polychoric corelation*

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. The items above the diagonal are the tetrachoric correlations, below the diagonal are raw correlations.

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

```
demo <- phi.demo() #compare the phi (lower off diagonal and polychoric correlations (uppe
#show the result from tetrachoric  which corrects for zero entries by default
round(demo$tetrachoric$rho,2)
#show the result from phi2poly
#tetrachorics above the diagonal, phi below the diagonal
round(demo$phis,2)
```

`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

requires the mvtnorm package

Value

a polychoric correlation

Author(s)

William Revelle

See Also

`tetrachoric`, `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 fa, irt.fa, VSS, ICLUST, omega, factor.pa, or principal.

Usage

```
plot.psych(x,labels=NULL,...)
plot.irt(x,xlab,ylab,main,D,type=c("ICC","IIC","test"),cut=.3,labels,keys=NULL,.
plot.poly(x,D,xlab,ylab,ylim,main,type=c("ICC","IIC","test"),cut=.3,labels,keys=
```

Arguments

x	The object to plot
labels	Variable labels
xlab	Label for the x axis – defaults to Latent Trait
ylab	Label for the y axis
ylim	Specify the limits for the y axis
main	Main title for graph
type	ICC plots items, IIC plots item information, test plots test information, defaults to IIC for plot.irt, to ICC for plot.poly
D	The discrimination parameter
cut	Only plot item responses with discrimiantion greater than cut
keys	Used in plotting irt results from irt.fa.
...	other calls to plot

Details

Passes the appropriate values to plot. For plotting the results of irt.fa, there are three options: type = "ICC" (default) will plot the item characteristic respone function. type = "IIC" will plot the item information function, and type= "test" will plot the test information function.

These are calls to the generic plot function that are intercepted for objects of type "psych". More precise plotting control is available in the separate plot functions. plot may be used for psych objects returned from fa, irt.fa, ICLUST, omega, as well as principal

Objects from irt.fa are plotted according to "type" (Item informations, item characteristics, or test information). In addition, plots for selected items may be done if using the keys matrix. Plots of irt information return two invisible objects, the average area under the curve (the average information) for each item as well as where the item is most informative.

Value

Graphic output for factor analysis, cluster analysis and item response analysis.

Note

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

Author(s)

William Revelle

See Also

VSS.plot and factor.plot, cluster.plot, fa, irt.fa, VSS, ICLUST, omega, factor.pa, or principal

Examples

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

#not run
#data(bfi)
#e.irt <- irt.fa(bfi[11:15])  #just the extraversion items
#plot(e.irt)   #the information curves
#
#ic <- iclust(test.data,3)   #shows hierarchical structure
#plot(ic)                    #plots loadings
#
```

---

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

---

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 the polychoric 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

```
demo <- phi.demo() #compare the phi (lower off diagonal and polychoric correlations (uppe
#show the result from poly.mat
round(demo$tetrachoric$rho,2)
#show the result from phi2poly
#tetrachorics above the diagonal, phi below the diagonal
round(demo$phis,2)
```

predict.psych *Prediction function for factor analysis or principal components*

Description

Finds predicted factor/component scores from a factor analysis or components analysis of data set A predicted to data set B. Predicted factor scores use the weights matrix used to find estimated factor scores, predicted components use the loadings matrix.

Usage

```
predict.psych(object, data,old.data,...)
```

Arguments

object	the result of a factor analysis or principal components analysis of data set A
data	Data set B, of the same number of variables as data set A.
old.data	if specified, the data set B will be standardized in terms of values from the old data
...	More options to pass to predictions

Value

Predicted factor/components scores.

Note

Thanks to Reinhold Hatzinger for the suggestion and request

Author(s)

William Revelle

See Also

fa, principal

Examples

```
set.seed(42)
x <- sim.item(12,500)
f2 <- fa(x[1:250,],2,scores=TRUE)  # a two factor solution
p2 <- principal(x[1:250,],2,scores=TRUE)  # a two component solution
round(cor(f2$scores,p2$scores),2)  #correlate the components and factors from the A set
#find the predicted scores (The B set)
pf2 <- predict(f2,x[251:500,])
pp2 <- predict(p2,x[251:500,])
round(cor(pf2,pp2),2)   #find the correlations in the B set
#test how well these predicted scores match the factor scores from the second set
fp2 <- fa(x[251:500,],2,scores=TRUE)
round(cor(fp2$scores,pf2),2)
#note that the signs of the factors in the second set are arbitrary
```

`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=F
```

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", "quatimax", "promax", "oblimin", "simplimax", and "cluster" are possible rotations/transformations 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, find 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
`oblique.scores`	If TRUE (default), then the component scores are based upon the structure matrix. If FALSE, upon the pattern matrix.

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. To point this out, unrotated principal components are labelled as PCi, while rotated PCs are now labeled as RCi (for rotated components) and obliquely transformed components as TCi (for transformed components). (Thanks to Ulrike Gromping for this suggestion.)

Rotations and transformations are either part of psych (Promax and cluster), of base R (varimax), or of GPArotation (simplimax, quartimax, oblimin).

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 are found by regression where the regression weights are $R^{-1}\lambda$ where λ is the matrix of component loadings. The regression approach is done to be parallel with the factor analysis function `fa`. The regression weights are found from the inverse of the correlation matrix times the component loadings. This has the result that the component scores are standard scores (mean=0, sd = 1) of the standardized input. A comparison to the scores from `princomp` shows this difference. princomp does not, by default, standardize the data matrix, nor are the components themselves standardized. By default, the regression weights are found from the Structure matrix, not the Pattern matrix.

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
`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 $f = log(trace((FF'+U2)^{-1}R) - log(\|(FF'+U2)^{-1}R\|) - n.items$. Because components do not minimize the off diagonal, this fit will be not as good as for factor analysis.
`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
`weights`	The beta weights to find the principal components from the data

R2	The multiple R square between the factors and factor score estimates, if they were to be found. (From Grice, 2001) For components, these are of course 1.0.
valid	The correlations of the component score estimates with the components, if they were to be found and unit weights were used. (So called course coding).

Author(s)

William Revelle

References

Grice, James W. (2001), Computing and evaluating factor scores. Psychological Methods, 6, 430-450

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 (to test for the number of components or factors to extract), VSS.scree and fa.parallel to show a scree plot and compare it with random resamplings of the data), factor2cluster (for course coding keys), fa (for factor analysis), factor.congruence (to compare solutions)

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$cov,4,rotate="varimax")
mr <- fa(Harman74.cor$cov,4,rotate="varimax")  #minres factor analysis
pa <- fa(Harman74.cor$cov,4,rotate="varimax",fm="pa")  # principal axis factor analysis
round(factor.congruence(list(pc,mr,pa)),2)
```

---

print.psych *Print and summary functions for the psych class*

Description

Give limited output (print) or somewhat more detailed (summary) for most of the functions in psych.

Usage

```
print.psych(x,digits=2,all=FALSE,cut=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
cut	Cluster loadings < cut will not be printed. For the factor analysis functions (fa and factor.pa etc.), cut defaults to 0, for ICLUST 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, ask for the structure of the particular object: (str(theobject).

Alternatively, 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 ICLUSTfor 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:
 keys <- make.keys(25,keys.list,item.labels=colnames(bfi[1:25]))
 scores <- score.items(keys,bfi[1:25])
 scores
 summary(scores)
```

`Promax` *Perform promax or targeted rotations 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. target.rot does general target rotations to an arbitrary target matrix. The default target rotation is for an independent cluster solution.

Usage

```
Promax(x, m = 4)
target.rot(x,keys=NULL)
```

Arguments

`x`	A loadings matrix
`m`	the power to which to raise the varimax loadings (for Promax)
`keys`	An arbitrary target matrix, can be composed of any weights, but probably -1,0, 1 weights. If missing, the target is the independent cluster structure determined by assigning every item to it's highest loaded factor.

Details

Promax 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, `fa` or ealier (`factor.pa`, `factor.minres` or `principal`) functions and select just the loadings matrix for analysis.

The target.rot function is an adaptation of a function of Michael Browne's to do rotations to arbitrary target matrices. Suggested by Pat Shrout.

The default for target.rot is to rotate to an independent cluster structure (every items is assigned to a group with its highest loading.)

target.rot will not handle targets that have linear dependencies (e.g., a pure bifactor model where there is a g loading and a group factor for all variables).

Value

`loadings`	Oblique factor loadings
`rotmat`	The rotation matrix applied to the original loadings to produce the promax soluion or the targeted matrix
`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. Further modified to do targeted rotation similar to a function of Michael Browne.

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, factor.minres, or principal

Examples

```
jen <- sim.hierarchical()
f3 <- factor.minres(jen,3)
Promax(f3)
target.rot(f3)
m3 <- factanal(covmat=jen,factors=3)
Promax(m3)  #example of taking the output from factanal
#compare this rotation with the solution from a targeted rotation aimed for an independen
target.rot(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 (Williams's Test), or the difference between two dependent correlations with different variables (Steiger Tests).

Usage

```
r.test(n, r12, r34 = NULL, r23 = NULL, r13 = NULL, r14 = NULL, r24 = NULL, n2 =
```

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. Instead, he recommends Williams' test. (See also Dunn and Clark, 1971). 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.

Williams, E.J. (1959) Regression analysis. Wiley, New York, 1959.

See Also

See also `corr.test` which tests all the elements of a correlation matrix, and `cortest.mat` to compare two matrices of correlations. r.test 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 clipboard is easy but a bit obscure, particularly for Mac users. This is just an easier way to do so. Data may be copied to the clipboard from Exel spreadsheets, csv files, or fixed width formatted files and then into a data.frame. Data may also be read from lower (or upper) triangular matrices and filled out to square matrices.

Usage

```
read.clipboard(header = TRUE, ...)   #assumes headers and tab or space delimited
read.clipboard.csv(header=TRUE,sep=',',...)   #assumes headers and comma delimit
read.clipboard.tab(header=TRUE,sep='\t',...)   #assumes headers and tab delimite
read.clipboard.lower(diag=TRUE,names=FALSE,...)  #read in a matrix given the low
read.clipboard.upper(diag=TRUE,names=FALSE,...)
read.clipboard.fwf(header=FALSE,widths=rep(1,10),...)  #read in data using a fix
```

Arguments

`header`	Does the first row have variable labels
`sep`	What is the designated separater between data fields?
`diag`	for upper or lower triangular matrices, is the diagonal specified or not
`names`	for read.clipboard.lower or upper, are colnames in the the first column
`widths`	how wide are the columns in fixed width input. The default is to read 10 columns of size 1.
`...`	Other parameters to pass to read

Details

A typical session of R might involve data stored in text files, generated online, etc. Although it is easy to just read from a file (particularly if using file.choose(), 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.

If the input file that was copied into the clipboard was an Excel file with blanks for missing data, then read.clipboard.tab() will correctly replace the blanks with NAs. Similarly for a csv file with blank entries, read.clipboard.csv will replace empty fields with NA.

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.

Many articles will report lower triangular matrices with variable labels in the first column. read.clipboard.lower (or read.clipboard.upper) will handle this case as well.

read.clipboard.fwf will read fixed format files from the clipboard. It includes a patch to read.fwf which will not read from the clipboard or from remote file. See read.fwf for documentation of how to specify the widths.

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)
#my.matrix <- read.clipboard.lower()
```

---

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)
```

`reverse.code` *Reverse the coding of selected items prior to scale analysis*

Description

Some IRT functions require all items to be coded in the same direction. Some data sets have items that need to be reverse coded (e.g., 6 -> 1, 1 -> 6). reverse.code will flip items based upon a keys vector of 1s and -1s. Reversed items are subtracted from the item max + item min. These may be specified or may be calculated.

Usage

```
reverse.code(keys, items, mini = NULL, maxi = NULL)
```

Arguments

keys	A vector of 1s and -1s. -1 implies reverse the item
items	A data set of items
mini	if NULL, the empirical minimum for each item. Otherwise, a vector of minima
maxi	f NULL, the empirical maximum for each item. Otherwise, a vector of maxima

Details

Not a very complicated function, but useful in the case that items need to be reversed prior to using IRT functions from the ltm or eRM packages. Most psych functions do not require reversing prior to analysis, but will do so within the function.

Value

The corrected items.

Examples

```
original <- matrix(sample(6,50,replace=TRUE),10,5)
keys <- c(1,1,-1,-1,1)  #reverse the 3rd and 4th items
new <- reverse.code(keys,original,mini=rep(1,5),maxi=rep(6,5))
original[1:3,]
new[1:3,]
```

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`

`scatter.hist` *Draw a scatter plot with associated X and Y histograms, densitie and correlation*

Description

Draw a X Y scatter plot with associated X and Y histograms with estimated densities. Partly a demonstration of the use of layout. Also includes lowess smooth or linear model slope, as well as correlation. Adapted from addicted to R example 78

Usage

```
scatter.hist(x,y=NULL,smooth=TRUE,ab=FALSE,correl=TRUE,density=TRUE,ellipse=TRUE
```

Arguments

`x`	The X vector, or the first column of a data.frame or matrix.
`y`	The Y vector, of if X is a data.frame or matrix, the second column of X
`smooth`	if TRUE, then loess smooth it
`ab`	if TRUE, then show the best fitting linear fit
`correl`	TRUE: Show the correlation
`density`	TRUE: Show the estimated densities
`ellipse`	TRUE: draw 1 and 2 sigma ellipses and smooth
`digits`	How many digits to use if showing the correlation
`cex.cor`	Adjustment for the size of the correlation
`xlab`	Label for the x axis
`ylab`	Label for the y axis
`title`	An optional title
`...`	Other parameters for graphics

Details

Just a straightforward application of layout and barplot, with some tricks taken from `pairs.panels`. The various options allow for correlation ellipses (1 and 2 sigma from the mean), lowess smooths, linear fits, density curves on the histograms, and the value of the correlation. ellipse = TRUE implies smooth = TRUE)

Note

Adapted from Addicted to R example 78

Author(s)

William Revelle

See Also

`pairs.panels` for multiple plots, `multi.hist` for multiple histograms.

Examples

```
data(sat.act)
with(sat.act,scatter.hist(SATV,SATQ))
#or for something a bit more splashy
scatter.hist(sat.act[5:6],pch=(19+sat.act$gender),col=c("blue","red")[sat.act$gender])
```

Schmid — *12 variables created by Schmid and Leiman to show the Schmid-Leiman Transformation*

Description

John Schmid and John M. Leiman (1957) discuss how to transform a hierarchical factor structure to a bifactor structure. Schmid contains the example 12 x 12 correlation matrix. schmid.leiman is a 12 x 12 correlation matrix with communalities on the diagonal. This can be used to show the effect of correcting for attenuation. Two additional data sets are taken from Chen et al. (2006).

Usage

```
data(Schmid)
```

Details

Two artificial correlation matrices from Schmid and Leiman (1957). One real and one artificial covariance matrices from Chen et al. (2006).

- Schmid: a 12 x 12 artificial correlation matrix created to show the Schmid-Leiman transformation.
- schmid.leiman: A 12 x 12 matrix with communalities on the diagonal. Treating this as a covariance matrix shows the 6 x 6 factor solution
- Chen: An 18 x 18 covariance matrix of health related quality of life items from Chen et al. (2006). Number of observations = 403. The first item is a measure of the quality of life. The remaining 17 items form four subfactors: The items are (a) Cognition subscale: “Have difficulty reasoning and solving problems?" “React slowly to things that were said or done?"; “Become confused and start several actions at a time?" “Forget where you put things or appointments?"; “Have difficulty concentrating?" (b) Vitality subscale: “Feel tired?" “Have enough energy to do the things you want?" (R) “Feel worn out?" ; “Feel full of pep?" (R). (c) Mental health subscale: “Feel calm and peaceful?"(R) “Feel downhearted and blue?"; “Feel very happy"(R) ; “Feel very nervous?" ; “Feel so down in the dumps nothing could cheer you up? (d) Disease worry subscale: “Were you afraid because of your health?"; “Were you frustrated about your health?"; “Was your health a worry in your life?" .
- West: A 16 x 16 artificial covariance matrix from Chen et al. (2006).

Source

John Schmid Jr. and John. M. Leiman (1957), The development of hierarchical factor solutions.Psychometrika, 22, 83-90.

F.F. Chen, S.G. West, and K.H. Sousa.(2006) A comparison of bifactor and second-order models of quality of life. Multivariate Behavioral Research, 41(2):189-225, 2006.

References

Y.-F. Yung, D.Thissen, and L.D. McLeod. (1999) On the relationship between the higher-order factor model and the hierarchical factor model. Psychometrika, 64(2):113-128, 1999.

Examples

```
data(Schmid)
cor.plot(Schmid,TRUE)
print(fa(Schmid,6,rotate="oblimin"),cut=0)  #shows an oblique solution
round(cov2cor(schmid.leiman),2)
cor.plot(cov2cor(West),TRUE)
```

---

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, fm = "minres",digits=2,rotate="oblimin",n.obs=NA,opt
```

Arguments

model	A correlation matrix
nfactors	Number of factors to extract
fm	the default is to do minres. fm="pa" for principal axes, fm="pc" for principal components, fm = "minres" for minimum residual (OLS), pc="ml" 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
option	When asking for just two group factors, option can be for "equal", "first" or "second"
Phi	If Phi is specified, then the analysis is done on a pattern matrix with the associated factor intercorrelation (Phi) matrix. This allows for reanalysess of published results
...	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. There are three possible options for this condition: setting the general factor loadings between the two lower order factors to be "equal" which will be the sqrt(oblique correlations between the factors) or to "first" or "second" in which case the general factor is equated with either the first or second group factor. A message is issued suggesting that the model is not really well defined.

A diagnostic tool for testing the appropriateness of a hierarchical model is p2 which is the percent of the common variance for each variable that is general factor variance. In general, p2 should not have much variance.

Value

sl	loadings on g + nfactors group factors, communalities, uniqueness, percent of g2 of h2
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
if(!require(GPArotation)) {message("I am sorry, you must have GPArotation installed to us
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, Guttman lambda 6 and item whole correlations*

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, Guttman's Lambda 6, the average r, the scale intercorrelations, and the item by scale correlations (raw and corrected for item overlap). 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. If the input is a square matrix, then it is assumed that the input is a covariance or correlation matix and scores are not found, but the item statistics are reported. (Similar functionality to cluster.cor). response.frequencies reports the frequency of item endorsements fore each response category for polytomous or multiple choice items.

Usage

```
score.items(keys, items, totals = FALSE, ilabels = NULL,missing=TRUE, impute="me
response.frequencies(items,max=10)
```

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	missing = TRUE is the normal case and data are imputed according to the impute option. missing=FALSE, only complete cases are scored.
impute	impute="median" replaces missing values with the item median, impute = "mean" replaces values with the mean response. impute="none" the subject's scores are based upon the average of the keyed, but non missing scores.
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. Alternatively, in response frequencies, it is maximum number of alternative responses to count.
digits	Number of digits to report

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 as it would seem to make more sense to report scale scores in the metric of the item.

Various estimates of scale reliability include "Cronbach's alpha", Guttman's Lambda 6, 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.

Surprisingly, 106 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.

α (Cronbach, 1951) is the same as Guttman's $\lambda 3$ (Guttman, 1945) and may be found by

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

Perhaps because it is so easy to calculate and is available in most commercial programs, alpha 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 β (Revelle, 1979; see ICLUST) and ω_h (see omega) are more appropriate estimates of the general factor saturation. ω_t (see omega) is a better estimate of the reliability of the total test.

Guttman's Lambda 6 (G6) 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

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

The squared multiple correlation is a lower bound for the item communality and as the number of items increases, becomes a better estimate.

G6 is also sensitive to lumpyness in the test and should not be taken as a measure of unifactorial structure. For lumpy tests, it will be greater than alpha. For tests with equal item loadings, alpha >

G6, but if the loadings are unequal or if there is a general factor, G6 > alpha. Although it is normal when scoring just a single scale to calculate G6 from just those items within the scale, logically it is appropriate to estimate an item reliability from all items available. This is done here and is labeled as G6* to identify the subtle difference.

Alpha and G6* are both positive functions 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, raw alpha is sensitive to differences in the item variances. Standardized alpha is based upon the correlations rather than the covariances. alpha is a generalization of an earlier estimate of reliability for tests with dichotomous items developed by Kuder and Richardson, known as KR20, and a shortcut approximation, KR21. (See Revelle, in prep).

More complete reliability analyses of a single scale can be done using the `omega` function which finds ω_h and ω_t based upon a hierarchical factor analysis. Alternative estimates of the Greatest Lower Bound for the reliability are found in the `guttman` function.

Alpha is a poor estimate of the general factor saturation of a test (see Revelle and Zinbarg, 2009; 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.

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). For data with a great deal of missingness, yet another option is to just find the average of the available responses (impute="none"). This is useful for findings means for scales for the SAPA project where most scales are estimated from random sub samples of the items from the scale.

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.
`G6`	Guttman's Lambda 6 measure of reliability
`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)
`item.corrected`	The item by scale correlations for each item, corrected for item overlap by replacing the item variance with the smc for that item
`response.freq`	The response frequency (based upon number of non-missing responses) for each alternative.
`missing`	How many items were not answered for each scale

Note

It is important to recognize in the case of massively missing data (e.g., data from a Synthetic Apeture Personality Assessment study where perhaps only 10-50% of the items per scale are given to any one subject)) that the number of items per scale, and hence standardized alpha, is not the nominal value and hence alpha will be overestimated. For this case (impute="none"), an additional alpha (alpha.ob) is reported.

Author(s)

William Revelle

References

Revelle, W. (in preparation) An introduction to psychometric theory with applications in R. http://personality-project.org/r/book

Revelle W. and R.E. Zinbarg. (2009) Coefficients alpha, beta, omega and the glb: comments on Sijtsma. Psychometrika, 74(1):145-154.

Zinbarg, R. E., Revelle, W., Yovel, I. and Li, W. (2005) Cronbach's alpha, Revelle's beta, and McDonald's omega h, Their relations with each other and two alternative conceptualizations of reliability, Psychometrika, 70, 123-133.

See Also

make.keys for a convenient way to create the keys file, score.multiple.choice for multiple choice items,
alpha, correct.cor, cluster.cor, cluster.loadings, omega, guttman for item/scale analysis. In addition, the irt.fa function provides an alternative way of examining the structure of a test and emphasizes item response theory approaches to the information returned by each item and the total test.

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:
 keys <- make.keys(28,keys.list,item.labels=colnames(bfi))
 scores <- score.items(keys,bfi,min=1,max=6)
 summary(scores)
 #to get the response frequencies, we need to not use the age variable
 scores <- score.items(keys[1:27,],bfi[1:27],min=1,max=6)
 scores

 #compare this output to that for the impute="none" option.
 #first make many of the items missing
 missing.bfi <- bfi
 missing.bfi[sample(dim(bfi)[1],500),sample(dim(bfi)[2],10)] <- NA
 scores <- score.items(keys,missing.bfi,impute="none",min=1,max=6)
 summary(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, m
```

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
```

---

scrub — *A utility for basic data cleaning and recoding. Changes values outside of minimum and maximum limits to NA.*

Description

A tedious part of data analysis is addressing the problem of miscoded data that need to be converted to NA. For a given data.frame or matrix, scrub will set all values of columns from=from to to=to that are less than a set (vector) of min values or more than a vector of max values to NA. Can also be used to do basic recoding of data for all values=isvalue to newvalue.

Usage

```
scrub(x, where, min, max,isvalue,newvalue)
```

Arguments

x	a data frame or matrix
where	The variables to examine. (Can be by name or by column number)
min	a vector of minimum values that are acceptable
max	a vector of maximum values that are acceptable
isvalue	a vector of values to be converted to newvalue (one per variable)
newvalue	a vector of values to replace those that match isvalue

Details

Solves a tedious problem that can be done directly but that is sometimes awkward. Will either replace specified values with NA or

Value

The corrected data frame.

Note

Probably could be optimized to avoid one loop

Author(s)

William Revelle

See Also

reverse.code, rescale for other simple utilities.

Examples

```
data(attitude)
x <- scrub(attitude,isvalue =c(30,40,50), newvalue = c(930,940,950))  #will change all oc
x1 <- scrub(attitude, where=4, isvalue =c(30,40,50), newvalue = c(930,940,950)) #will jus
#get rid of a complicated set of cases and replace with missing values
y <- scrub(attitude,where=2:4,min=c(20,30,40),max= c(120,110,100),isvalue= c(32,43,54))
y
scrub(attitude,where="learning",isvalue=68,newvalue=999) #change a column by name
scrub(attitude,where="learning",min=45,newvalue=999) #change a column by name
```

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)    #deprecated
```

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 Because this problem seems to have been fixed, SD will be removed eventually.

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)
```

---

set.cor *Set Correlation and Multiple Regression from raw or matrix input*

Description

Finds Cohen's Set Correlation between a predictor set of variables (x) and a criterion set (y). Also finds multiple correlations between x variables and each of the y variables. Will work with either raw data or a correlation matrix. A set of covariates (z) can be partialled from the x and y sets.

Usage

```
set.cor(y, x,data, z=NULL,n.obs=NULL,use="pairwise")
mat.regress(y, x,data, z=NULL,n.obs=NULL)
```

Arguments

data	a matrix or data.frame of correlations or, if not square, of raw data
y	either the column numbers of the y set (e.g., c(2,4,6) or the column names of the y set (e.g., c("Flags","Addition")
x	either the column numbers of the x set (e.g., c(1,3,5) or the column names of the x set (e.g. c("Cubes","PaperFormBoard")
n.obs	If specified, then confidence intervals, etc. are calculated, not needed if raw data are given
z	the column names or numbers of the set of covariates
use	find the correlations "pairwise" (default) or just use "complete" cases (to match the lm function)

Details

Cohen (1982) introduced the set correlation, a multivariate generalization of the multiple correlation to measure the overall relationship between two sets of variables. It is an application of canoncial correlation (Hotelling, 1936) and $1 - \prod(1 - \rho_i^2)$ where ρ_i^2 is the squared canonical correlation. Set correlation is the amount of shared variance (R2) between two sets of variables. With the addition of a third, covariate set, set correlation will find multivariate R2, as well as partial and semi partial R2. (The semi and bipartial options are not yet implemented.) Details on set correlation may be found in Cohen (1982) Cohen, Cohen, Aiken and West (2003).

Unfortunately, the R2 is sensitive to one of the canonical correlations being very high. An alternative, T2, is the proportion of additive variance and is the average of the squared canonicals. (Cohen et al., 2003), see also Cramer and Nicewander (1979). This average, because it includes some very small canonical correlations, will tend to be too small. Cohen et al. admonition is appropriate: "In the final analysis, however, analysts must be guided by their substantive and methodological conceptions of the problem at hand in their choice of a measure of association." (p613).

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

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

An additional output is the R2 found using Cohen's set correlation (Cohen, 1982). This is a measure of how much variance and the x and y set share.

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 is rectangular, correlations are found from the data.

The print function reports t and p values for the beta weights, the summary function just reports the beta weights.

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
`setR2`	The multiple R2 of the set correlation between the x and y sets

.

Note

As of April 30, 2011, the order of x and y was swapped in the call to be consistent with the general y ~ x syntax of the lm and aov functions. In addition, the primary name of the function was switched to set.cor from mat.regress to reflect the estimation of the set correlation.

The denominator degrees of freedom for the set correlation does not match that reported by Cohen et al., 2003 in the example on page 621 but does match the formula on page 615, except for the typo in the estimation of F (see Cohen 1982). The difference seems to be that they are adding in a correction factor of df 2 = df2 + df1.

Author(s)

William Revelle

Maintainer: William Revelle <revelle@northwestern.edu>

References

J. Cohen (1982) Set correlation as a general mulitivariate data-analytic method. Multivariate Behavioral Research, 17(3):301-341.

J. Cohen, P. Cohen, S.G. West, and L.S. Aiken. (2003) Applied multiple regression/correlation analysis for the behavioral sciences. L. Erlbaum Associates, Mahwah, N.J., 3rd ed edition.

H. Hotelling. (1936) Relations between two sets of variates. Biometrika 28(3/4):321-377.

E.Cramer and W A. Nicewander (1979) Some symmetric, invariant measures of multivariate association. Psychometrika, 44:43-54.

See Also

cluster.cor, factor2cluster,principal,ICLUST, link{cancor} and cca in the yacca package.

Examples

```
#the Kelly data from Hoteling
kelly <- structure(list(speed = c(1, 0.4248, 0.042, 0.0215, 0.0573), power = c(0.4248,
1, 0.1487, 0.2489, 0.2843), words = c(0.042, 0.1487, 1, 0.6693,
0.4662), symbols = c(0.0215, 0.2489, 0.6693, 1, 0.6915), meaningless = c(0.0573,
0.2843, 0.4662, 0.6915, 1)), .Names = c("speed", "power", "words",
"symbols", "meaningless"), class = "data.frame", row.names = c(NA,
-5L))

kelly

set.cor(1:2,3:5,kelly)

set.cor(y=c(7:9),x=c(1:6),data=Thurstone,n.obs=213)
#now try partialling out some variables
set.cor(y=c(7:9),x=c(1:3),z=c(4:6),data=Thurstone) #compare with the previous

set.cor(x=c("gender","education"),y=c("SATV","SATQ"),data=sat.act) # regression from raw
```

sim *Functions to simulate psychological/psychometric data.*

Description

A number of functions in the psych package will generate simulated data. These functions include sim for a factor simplex, and sim.simplex for a data simplex, sim.circ for a circumplex structure, sim.congeneric for a one factor factor congeneric model, sim.dichot to simulate dichotomous items, sim.hierarchical to create a hierarchical factor model, sim.item a more general item simulation, sim.minor to simulate major and minor factors, sim.omega to test various examples of omega, sim.parallel to compare the efficiency of various ways of deterimining the number of factors, sim.rasch to create simulated rasch data, sim.irt to create general 1 to 4 parameter IRT data by calling sim.npl 1 to 4 parameter logistic IRT or sim.npn 1 to 4 paramater normal IRT, sim.structural a general simulation of structural models, and sim.anova for ANOVA and lm simulations, and sim.VSS. Some of these functions are separately documented and are listed here for ease of the help function. See each function for more detailed help.

Usage

```
sim(fx=NULL,Phi=NULL,fy=NULL,alpha=.8,lambda = 0,n=0,mu=NULL,raw=TRUE)
sim.simplex(nvar =12, alpha=.8,lambda=0,beta=1,mu=NULL, n=0)
sim.minor(nvar=12,nfact=3,n=0,fbig=NULL,fsmall = c(-.2,.2),bipolar=TRUE)
sim.omega(nvar=12,nfact=3,n=0,fbig=NULL,fsmall = c(-.2,.2),bipolar=TRUE,om.fact=
sim.parallel(ntrials=10,nvar = c(12,24,36,48),nfact = c(1,2,3,4,6),
n = c(200,400))
sim.rasch(nvar = 5,n = 500, low=-3,high=3,d=NULL, a=1,mu=0,sd=1)
sim.irt(nvar = 5, n = 500, low=-3, high=3,a=NULL,c=0,z=1,d=NULL,mu=0,sd=1,mod="l
sim.npl(nvar = 5, n = 500, low=-3,high=3,a=NULL,c=0,z=1,d=NULL,mu=0,sd=1)
sim.npn(nvar = 5, n = 500, low=-3,high=3,a=NULL,c=0,z=1,d=NULL,mu=0,sd=1)
```

Arguments

fx	The measurement model for x. If NULL, a 4 factor model is generated
Phi	The structure matrix of the latent variables
fy	The measurement model for y
mu	The means structure for the fx factors
n	Number of cases to simulate. If n=0 or NULL, the population matrix is returned.
raw	if raw=TRUE, raw data are returned as well.
nvar	Number of variables for a simplex structure
nfact	Number of large factors to simulate in sim.minor
alpha	the base correlation for an autoregressive simplex
lambda	the trait component of a State Trait Autoregressive Simplex
beta	Test reliability of a STARS simplex
fbig	Factor loadings for the main factors. Default is a simple structure with loadings sampled from (.8,.6) for nvar/nfact variables and 0 for the remaining. If fbig is specified, then each factor has loadings sampled from it.

bipolar	if TRUE, then positive and negative loadings are generated from fbig
om.fact	Number of factors to extract in omega
flip	In omega, should item signs be flipped if negative
option	In omega, for the case of two factors, how to weight them?
fsmall	nvar/2 small factors are generated with loadings sampled from (-.2,0,.2)
ntrials	Number of replications per level
low	lower difficulty for sim.rasch or sim.irt
high	higher difficulty for sim.rasch or sim.irt
a	if not specified as a vector, the descrimination parameter a = α will be set to 1.0 for all items
d	if not specified as a vector, item difficulties (d = δ) will range from low to high
c	the gamma parameter: if not specified as a vector, the guessing asymptote is set to 0
z	the zeta parameter: if not specified as a vector, set to 1
sd	the standard deviation for the underlying latent variable in the irt simulations
mod	which IRT model to use, mod="logistic" simulates a logistic function, otherwise, a normal function

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. For a much longer discussion of the use of simulation in psychometrics, see the accompany vignettes.

The default values for sim.structure is to generate a 4 factor, 12 variable data set with a simplex structure between the factors. This, and the simplex of items (sim.simplex) can also be converted in a STARS model with an autoregressive component (alpha) and a stable trait component (lambda).

Two data structures that are particular challenges to exploratory factor analysis are the simplex structure and the presence of minor factors. Simplex structures sim.simplex will typically occur in developmental or learning contexts and have a correlation structure of r between adjacent variables and r^n for variables n apart. Although just one latent variable (r) needs to be estimated, the structure will have nvar-1 factors.

An alternative version of the simplex is the State-Trait-Auto Regressive Structure (STARS) which has both a simplex state structure, with autoregressive path alpha and a trait structure with path lambda. This simulated in sim.simplex by specifying a non-zero lambda value.

Many simulations of factor structures assume that except for the major factors, all residuals are normally distributed around 0. An alternative, and perhaps more realistic situation, is that the there are a few major (big) factors and many minor (small) factors. The challenge is thus to identify the major factors. sim.minor generates such structures. The structures generated can be thought of as havinga a major factor structure with some small correlated residuals.

Although coefficient

$$\omega$$

is a very useful indicator of the general factor saturation of a unifactorial test (one with perhaps several sub factors), it has problems with the case of multiple, independent factors. In this situation, one of the factors is labelled as "general" and the omega estimate is too large. This situation may be explored using the sim.omega function.

The four irt simulations, sim.rasch, sim.irt, sim.npl and sim.npn, simulate dichotomous items following the Item Response model. sim.irt just calls either sim.npl (for logistic models) or sim.npn (for normal models) depending upon the specification of the model.

The logistic model is

$$P(i,j) = \gamma + \frac{\zeta - \gamma}{1 + e^{\alpha(\delta - \theta)}}$$

where γ is the lower asymptote or guesssing parameter, ζ is the upper asymptote (normally 1), α is item discrimination and δ is item difficulty. For the 1 Paramater Logistic (Rasch) model, gamma=0, zeta=1, alpha=1 and item difficulty is the only free parameter to specify.

For the 2PL and 2PN models, a = α and d = δ are specified.
For the 3PL or 3PN models, items also differ in their guessing parameter c =γ.
For the 4PL and 4PN models, the upper asymptote, z= ζ is also specified.
(Graphics of these may be seen in the demonstrations for the `logistic` function.)

The normal model (irt.npn calculates the probability using pnorm instead of the logistic function used in irt.npl, but the meaning of the parameters are otherwise the same. With the a = α parameter = 1.702 in the logitistic model the two models are practically identical.

Other simulation functions in psych are:

`sim.structure` A function to combine a measurement and structural model into one data matrix. Useful for understanding structural equation models. Combined with `structure.diagram` to see the proposed structure.

`sim.congeneric` A function to create congeneric items/tests for demonstrating classical test theory. This is just a special case of sim.structure.

`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.

`sim.minor` Create a factor structure for nvar variables defined by nfact major factors and nvar/2 "minor" factors for n observations.

Although the standard factor model assumes that K major factors (K « nvar) will account for the correlations among the variables

$$R = FF' + U^2$$

where R is of rank P and F is a P x K matrix of factor coefficients and U is a diagonal matrix of uniquenesses. However, in many cases, particularly when working with items, there are many small factors (sometimes referred to as correlated residuals) that need to be considered as well. This leads to a data structure such that

$$R = FF' + MM' + U^2$$

where R is a P x P matrix of correlations, F is a P x K factor loading matrix, M is a P x P/2 matrix of minor factor loadings, and U is a diagonal matrix (P x P) of uniquenesses.

Such a correlation matrix will have a poor χ^2 value in terms of goodness of fit if just the K factors are extracted, even though for all intents and purposes, it is well fit.

sim.minor will generate such data sets with big factors with loadings of .6 to .8 and small factors with loadings of -.2 to .2. These may both be adjusted.

`sim.parallel` Create a number of simulated data sets using sim.minor to show how parallel analysis works. The general observation is that with the presence of minor factors, parallel analysis

is probably best done with component eigen values rather than factor eigen values, even when using the factor model.

sim.anova Simulate a 3 way balanced ANOVA or linear model, with or without repeated measures. Useful for teaching research methods and generating teaching examples.

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

```
simplex <- sim.simplex() #create the default structure
round(simplex,2)  #the correlation matrix



congeneric <- sim.congeneric()
round(congeneric,2)
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)
R <- sim.structure(fx,Phi,fy)
cor.plot(R$model) #show it graphically

simp <- sim.simplex()
#show the simplex structure using cor.plot
cor.plot(simp,colors=TRUE,main="A simplex structure")
#Show a STARS model
simp <- sim.simplex(alpha=.8,lambda=.8)
#show the simplex structure using cor.plot
cor.plot(simp,colors=TRUE,main="State Trait Auto Regressive Simplex" )
```

---

sim.anova *Simulate a 3 way balanced ANOVA or linear model, with or without repeated measures.*

Description

For teaching basic statistics, it is useful to be able to generate examples suitable for analysis of variance or simple linear models. sim.anova will generate the design matrix of three independent variables (IV1, IV2, IV3) with an arbitrary number of levels and effect sizes for each main effect and interaction. IVs can be either continuous or categorical and can have linear or quadratic effects.

Either a single dependent variable or multiple (within subject) dependent variables are generated according to the specified model. The repeated measures are assumed to be tau equivalent with a specified reliability.

Usage

```
sim.anova(es1 = 0, es2 = 0, es3 = 0, es12 = 0, es13 = 0,
    es23 = 0, es123 = 0, es11=0,es22=0, es33=0,n = 2,n1 = 2, n2 = 2, n3 = 2, wit
```

Arguments

es1	Effect size of IV1
es2	Effect size of IV2
es3	Effect size of IV3
es12	Effect size of the IV1 x IV2 interaction
es13	Effect size of the IV1 x IV3 interaction
es23	Effect size of the IV2 x IV3 interaction
es123	Effect size of the IV1 x IV2 * IV3 interaction
es11	Effect size of the quadratric term of IV1
es22	Effect size of the quadratric term of IV2
es33	Effect size of the quadratric term of IV3
n	Sample size per cell (if all variables are categorical) or (if at least one variable is continuous), the total sample size
n1	Number of levels of IV1 (0) if continuous
n2	Number of levels of IV2
n3	Number of levels of IV3
within	if not NULL, then within should be a vector of the means of any repeated measures.
r	the correlation between the repeated measures (if they exist). This can be thought of as the reliablility of the measures.
factors	report the IVs as factors rather than numeric
center	center=TRUE provides orthogonal contrasts, center=FALSE adds the minimum value + 1 to all contrasts
std	Standardize the effect sizes by standardizing the IVs

Details

A simple simulation for teaching about ANOVA, regression and reliability. A variety of demonstrations of the relation between anova and lm can be shown.

The default is to produce categorical IVs (factors). For more than two levels of an IV, this will show the difference between the linear model and anova in terms of the comparisons made.

The within vector can be used to add congenerically equivalent dependent variables. These will have intercorrelations (reliabilities) of r and means as specified as values of within.

To demonstrate the effect of centered versus non-centering, make factors = center=FALSE. The default is to center the IVs. By not centering them, the lower order effects will be incorrect given the higher order interaction terms.

Value

y.df is a data.frame of the 3 IV values as well as the DV values.

`IV1 ... IV3` Independent variables 1 ... 3

`DV` If there is a single dependent variable

`DV.1 ... DV.n`
If within is specified, then the n within subject dependent variables

Author(s)

William Revelle

See Also

The general set of simulation functions in the psych package `sim`

Examples

```
set.seed(42)
data.df <- sim.anova(es1=1,es2=.5,es13=1)  # one main effect and one interaction
describe(data.df)
pairs.panels(data.df)   #show how the design variables are orthogonal
#
summary(lm(DV~IV1*IV2*IV3,data=data.df))
summary(aov(DV~IV1*IV2*IV3,data=data.df))
set.seed(42)
data.df <- sim.anova(es1=1,es2=.5,es13=1,center=FALSE)  # demonstrate the effect of not c
describe(data.df)
#
summary(lm(DV~IV1*IV2*IV3,data=data.df)) #this one is incorrect, because the IVs are not
summary(aov(DV~IV1*IV2*IV3,data=data.df)) #compare with the lm model
#now examine multiple levels and quadratic terms
set.seed(42)
data.df <- sim.anova(es1=1,es13=1,n2=3,n3=4,es22=1)
summary(lm(DV~IV1*IV2*IV3,data=data.df))
summary(aov(DV~IV1*IV2*IV3,data=data.df))
pairs.panels(data.df)
#
data.df <- sim.anova(es1=1,es2=-.5,within=c(-1,0,1),n=10)
pairs.panels(data.df)
```

---

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
categorical	continuous or categorical (discrete) variables.
low	values less than low are forced to low
high	values greater than high are forced to high

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, fa for an example of factor scores, irt.fa and polychoric for the treatment of item data with discrete values.

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$observed),2) # show  a congeneric correlation matrix
f1=fa(test$observed,scores=TRUE)
round(cor(f1$scores,test$latent),2)  #factor score estimates are correlated with but not
set.seed(42)
items <- sim.congeneric(N=500,short=FALSE,low=-2,high=2,categorical=TRUE) #500 responses
d4 <- irt.fa(items$observed)  #item response analysis of congeneric measures
```

---

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,mu = NULL)
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 factors
n	Number of subjects to generate: N=0 => population values
raw	raw=TRUE, report the raw data, raw=FALSE, report the sample correlation matrix.
mu	means for the individual variables

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 for ways of analyzing these data. Also see sim.structure to simulate a variety of structural models (e.g., multiple correlated factor models). The simulation uses the mvrnorm function from the MASS package.

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)

#simulate a non-hierarchical structure
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
gload <- matrix(rep(0,5))
five.factor <- sim.hierarchical(gload,fload,500,TRUE) #create sample data set
#do it again with a hierachical structure
gload <- matrix(rep(.7,5)  )
five.factor.g <- sim.hierarchical(gload,fload,500,TRUE) #create sample data set
#compare these two with omega
#not run
#om.5 <- omega(five.factor$observed,5)
#om.5g <- omega(five.factor.g$observed,5)
```

`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.

Usage

```
sim.item(nvar = 72, nsub = 500, circum = FALSE, xloading = 0.6, yloading = 0.6,
sim.circ(nvar = 72, nsub = 500, circum = TRUE, xloading = 0.6, yloading = 0.6, g
sim.dichot(nvar = 72, nsub = 500, circum = FALSE, xloading = 0.6, yloading = 0.6
item.dichot(nvar = 72, nsub = 500, circum = FALSE, xloading = 0.6, yloading = 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).

It is useful to compare the results of sim.item with sim.hierarchical. sim.item will produce a general factor that runs through all the items as well as two orthogonal factors. This produces a data set that is hard to represent with standard rotation techniques. Extracting 3 factors without rotation and then rotating the 2nd and 3rd factors reproduces the correct solution. But simple oblique rotation of 3 factors, or an omega analysis do not capture the underlying structure. See the last example.

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)$loadings,main="Circumplex Structure") #circumplex stru
#
#
plot(factor.pa(item.sim(16,500),2)$loadings,main="Simple Structure") #simple structure
#
cluster.plot(factor.pa(item.dichot(16,low=0,high=1),2))

#not run
#x12 <- sim.item(12,gloading=.6)
#f3 <- fa(x12,3,rotate="none")
#f3  #observe the general factor
#oblimin(f3$loadings[,2:3])  #show the 2nd and 3 factors.
#f3 <- fa(x12,3)   #now do it with oblimin rotation
#f3  # not what one naively expect.
```

`sim.structure` *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 with the correlation structure between the two sets.

Usage

```
sim.structural(fx=NULL,Phi=NULL,fy=NULL,f=NULL,n=0,raw=FALSE)
make.structural(fx=NULL,Phi=NULL,fy=NULL,f=NULL,n=0,raw=FALSE)   #deprecated
```

Arguments

`fx`	The measurement model for x
`Phi`	The structure matrix of the latent variables
`fy`	The measurement model for y
`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, fx and the structure model Phi, 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. structure.list and structure.list for making symbolic structures.

Examples

```
fx <-matrix(c( .9,.8,.6,rep(0,4),.6,.8,-.7),ncol=2)
fy <- matrix(c(.6,.5,.4),ncol=1)
rownames(fx) <- c("V","Q","A","nach","Anx")
rownames(fy)<- c("gpa","Pre","MA")
Phi <-matrix( c(1,0,.7,.0,1,.7,.7,.7,1),ncol=3)
gre.gpa <- sim.structural(fx,Phi,fy)
print(gre.gpa,2)
round(correct.cor(gre.gpa$model,gre.gpa$reliability),2)  #correct for attenuation to see

congeneric <- sim.structural(f=c(.9,.8,.7,.6)) # a congeneric model
congeneric
```

---

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=
```

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))
circ.sim.plot(x.df)
```

Arguments

samplesize	a vector of sample sizes to simulate
numberofvariables	vector of the number of variables to simulate
x.df	A data frame resulting from simulation.circ

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 nonzero 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
```

`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,covar=FALSE)
```

Arguments

`R`	A correlation matrix or a dataframe. In the latter case, correlations are found.
`covar`	if covar = TRUE and R is either a covariance matrix or data frame, then return the smc * variance for each item

Value

a vector of squared multiple correlations. Or, if covar=TRUE, a vector of squared multiple correlations * the item variances

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.diagram` *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). If the Rx or Ry matrices are specified, show the correlations between the x variables, or y variables.

Perhaps even more usefully, the function returns a model appropriate for running directly in the *sem package* written by John Fox. For this option to work directly, it is necessary to specfy that errrors=TRUE.

Input can be specified as matrices or the output from `fa`, `factor.pa`, factanal, 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. (This functionality is now included in structure.diagram.)

structure.diagram will draw the diagram without using Rgraphviz and is probably the preferred option. structure.graph will be removed eventually.

lavaan.diagram will draw either cfa or sem results from the lavaan package (> .4.0)

Usage

```
structure.diagram(fx, Phi=NULL,fy=NULL,labels=NULL,cut=.3,errors=FALSE,simple=TR
structure.graph(fx,  Phi = NULL,fy = NULL, out.file = NULL, labels = NULL, cut =
structure.sem(fx,  Phi = NULL, fy = NULL,out.file = NULL, labels = NULL, cut = 0
lavaan.diagram(fit,title,...)
```

Arguments

`fx`	a factor model on the x variables.
`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.
`fy`	a factor model on the y variables (can be empty)
`Rx`	The correlation matrix among the x variables
`Ry`	The correlation matrix among the y variables
`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
`fit`	The output from a lavaan cfa or sem
`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)
lr	Direction of diagram is from left to right (lr=TRUE, default) or from bottom to top (lr=FALSE)
e.size	size of the ellipses in structure.diagram
main	main title of diagram
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

The recommended function is structure.diagram which does not use Rgraphviz but which does not produce dot code either.

All three function return a matrix of commands suitable for using in the sem package. (Specify errors=TRUE to get code that will run directly in the sem package.)

The structure.graph output can be directed to an output file for post processing using the dot graphic language but requires that Rgraphviz is installed.

The figure is organized to show the appropriate paths between:

The correlations between the X variables (if Rx is specified)
The X variables and their latent factors (if fx is specified)
The latent X and the latent Y (if Phi is specified)
The latent Y and the observed Y (if fy is specified)
The correlations between the Y variables (if Ry is specified)

A confirmatory factor model would specify just fx and Phi, a structural model would include fx, Phi, and fy. The raw correlations could be shown by just including Rx and Ry.

lavaan.diagram may be called from the diagram function which also will call fa.diagram, omega.diagram or iclust.diagram, depending upon the class of the fit.

Other diagram functions include fa.diagram, omega.diagram. All of these functions use the various dia functions such as dia.rect, dia.ellipse, dia.arrow, dia.curve, dia.curved.arrow, and dia.shape.

Value

sem	(invisible) a model matrix (partially) ready for input to John Fox's sem package. It is of class "mod" for prettier output.
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 to create artificial data sets with particular structural properties.

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)
f1 <- structure.diagram(fx,Phi,fy,main="A structural path diagram")

#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"
f2 <- structure.diagram(X2,Phi=phi2,errors=FALSE,main="A symbolic model")

#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"
f3 <- structure.diagram(X2,Phi=phi2,main="an alternative representation")

#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")
f4 <- structure.diagram(X6,phi21,Y3)



# and finally, a regression model
X7 <- matrix(c("a","b","c","d","e","f"),nrow=6)
f5 <- structure.diagram(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)
f6<- structure.diagram(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. Factors with NULL values are filled with 0s.

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 for drawing it, or sim.structure for creating this data structure.

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, count=NULL,labs = NULL)
```

Arguments

x	A two dimensional table of counts with row and column names that can be converted to numeric values.
count	if present, then duplicate each row count times
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.

This function may also be used to convert an item response pattern table into a data table. e.g., the Bock data set `bock`.

Value

A matrix (or data.frame) of sum(x) rows and two columns.

Author(s)

William Revelle

See Also

`cubits` and `bock` data sets

Examples

```
data(cubits)
cubit <- table2matrix(cubits,labs=c("height","cubit"))
describe(cubit)
ellipses(cubit,n=1)
data(bock)
responses <- table2df(bock.table[,2:6],count=bock.table[,7],labs= paste("lsat6.",1:5,sep=
describe(responses)
```

---

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()
```

---

tetrachoric *Tetrachoric, polychoric, biserial and polyserial correlations from various types of input*

Description

The tetrachoric correlation is the inferred Pearson Correlation from a two x two table with the assumption of bivariate normality. The polychoric correlation generalizes this to the n x m table. Particularly important when doing Item Response Theory or converting comorbidity statistics using normal theory to correlations. Input may be a 2 x 2 table of cell frequencies, a vector of cell frequencies, or a data.frame or matrix of dichotomous data (for tetrachoric) or of numeric data (for polychoric). The biserial correlation is between a continuous y variable and a dichotmous x variable, which is assumed to have resulted from a dichotomized normal variable. Biserial is a special case

of the polyserial correlation, which is the inferred latent correlation between a continuous variable (X) and a ordered categorical variable (e.g., an item response). Input for these later two are data frames or matrices.

Usage

```
tetrachoric(x,correct=TRUE)
polychoric(x,polycor=FALSE, ML = FALSE, std.err=FALSE)
biserial(x,y)
polyserial(x,y)
poly.mat(x, short = TRUE, std.err = FALSE, ML = FALSE) #deprecated  use polychor
```

Arguments

x	The input may be in one of four forms: a) a data frame or matrix of dichotmous data (e.g., the lsat6 from the bock data set) or discrete numerical (i.e., not too many levels, e.g., the big 5 data set, bfi) for polychoric, or continuous for the case of biserial and polyserial. b) a 2 x 2 table of cell counts or cell frequencies (for tetrachoric) c) a vector with elements corresponding to the four cell frequencies (for tetrachoric) d) a vector with elements of the two marginal frequencies (row and column) and the comorbidity (for tetrachoric)
y	A (matrix or dataframe) of discrete scores. In the case of tetrachoric, these should be dichotomous, for polychoric not too many levels, for biserial they should be discrete (e.g., item responses) with not too many (<10?) categories.
correct	Correct for continuity in the case of zero entry cell for tetrachoric
polycor	If polycor=TRUE and the polycor package is installed, then use it when finding the polychoric correlations.
short	short=TRUE, just show the correlations, short=FALSE give the full hetcor output from John Fox's hetcor function if installed and if doing polychoric
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 — very slow!

Details

Tetrachoric correlations infer a latent Pearson correlation from a two x two table of frequencies with the assumption of bivariate normality. The estimation procedure is two stage ML. Cells with zero counts are replaced with .5 as a correction for continuity (correct=TRUE).

The data typically will be a raw data matrix of responses to a questionnaire scored either true/false (tetrachoric) or with a limited number of responses (polychoric). In both cases, the marginal frequencies are converted to normal theory thresholds and the resulting table for each item pair is converted to the (inferred) latent Pearson correlation that would produce the observed cell frequencies with the observed marginals. (See draw.tetra for an illustration.)

The tetrachoric correlation is used in a variety of contexts, one important one being in Item Response Theory (IRT) analyses of test scores, a second in the conversion of comorbity statistics to correlation coefficients. It is in this second context that examples of the sensitivity of the coefficient to the cell frequencies becomes apparent:

Consider the test data set from Kirk (1973) who reports the effectiveness of a ML algorithm for the tetrachoric correlation (see examples).

Examples include the lsat6 and lsat7 data sets in the bock data.

The polychoric function forms matrices of polychoric correlations by either using John Fox's polychor function or by an local function (polyc) and will also report the tau values for each alternative.

polychoric replaces poly.mat and is recommended. poly.mat is an alternative wrapper to the polycor function.

biserial and polyserial correlations are the inferred latent correlations equivalent to the observed point-biserial and point-polyserial correlations (which are themselves just Pearson correlations).

The polyserial function is meant to work with matrix or dataframe input and treats missing data by finding the pairwise Pearson r corrected by the overall (all observed cases) probability of response frequency. This is particularly useful for SAPA procedures with large amounts of missing data and no complete cases.

Ability tests and personality test matrices will typically have a cleaner structure when using tetrachoric or polychoric correlations than when using the normal Pearson correlation.

A biserial correlation (not to be confused with the point-biserial correlation which is just a Pearson correlation) is the latent correlation between x and y where y is continuous and x is dichotomous but assumed to represent an (unobserved) continuous normal variable. Let p = probability of x level 1, and q = 1 - p. Let zp = the normal ordinate of the z score associated with p. Then, $rbi = rs * \sqrt{(pq)}/zp$.

The 'ad hoc' polyserial correlation, rps is just $r = r * sqrt(n-1)/n)\sigma y/\sum(zpi)$ where zpi are the ordinates of the normal curve at the normal equivalent of the cut point boundaries between the item responses. (Olsson, 1982)

All of these were inspired by (and adapted from) John Fox's polychor package which should be used for precise ML estimates of the correlations. See, in particular, the hetcor function in the polychor package.

Value

rho	The (matrix) of tetrachoric/polychoric/biserial correlations
tau	The normal equivalent of the cutpoints

Note

For tetrachoric, in the degenerate case of a cell entry with zero observations, a correction for continuity is applied and .5 is added to the cell entry. A warning is issued. If correct=FALSE the correction is not applied.

Author(s)

William Revelle

References

A. Gunther and M. Hofler. Different results on tetrachorical correlations in mplus and stata-stata announces modified procedure. Int J Methods Psychiatr Res, 15(3):157-66, 2006.

David Kirk (1973) On the numerical approximation of the bivariate normal (tetrachoric) correlation coefficient. Psychometrika, 38, 259-268.

U.Olsson, F.Drasgow, and N.Dorans (1982). The polyserial correlation coefficient. Psychometrika, 47:337-347.

See Also

See also the polychor function in the polycor package. irt.fa uses the tetrachoric function to do item analysis with the fa factor analysis function. draw.tetra shows the logic behind a tetrachoric correlation (for teaching purpuses.)

Examples

```
if(require(mvtnorm)) {
data(bock)
tetrachoric(lsat6)
polychoric(lsat6)  #values should be the same
tetrachoric(matrix(c(44268,193,14,0),2,2))  #MPLUS reports.24
tetrachoric(matrix(c(44268,193,14,0),2,2),FALSE)  #Do not apply continuity correction --
tetrachoric(matrix(c(61661,1610,85,20),2,2)) #Mplus reports .35
tetrachoric(matrix(c(62503,105,768,0),2,2)) #Mplus reports -.10
tetrachoric(matrix(c(24875,265,47,0),2,2)) #Mplus reports  0
tetrachoric(matrix(c(24875,265,47,0),2,2),FALSE) #Do not apply continuity correction- com
tetrachoric(c(0.02275000, 0.0227501320, 0.500000000))
tetrachoric(c(0.0227501320, 0.0227501320, 0.500000000)) } else {message("Sorry, you must

# 4 plots comparing biserial to
set.seed(42)
x.4 <- sim.congeneric(loads =c(.9,.6,.3,0),N=1000,short=FALSE)
y  <- x.4$latent[,1]
for(i in 1:4) {
x <- x.4$observed[,i]
r <- round(cor(x,y),1)
ylow <- y[x<= 0]
yhigh <- y[x > 0]
yc <- c(ylow,yhigh)
rpb <- round(cor((x>=0),y),2)
rbis <- round(biserial(y,(x>=0)),2)
ellipses(x,y,ylim=c(-3,3),xlim=c(-4,3),pch=21 - (x>0),main =paste("r = ",r,"rpb = ",rpb,"

dlow <- density(ylow)
dhigh <- density(yhigh)
points(dlow$y*5-4,dlow$x,typ="l",lty="dashed")
lines(dhigh$y*5-4,dhigh$x,typ="l")
}
```

---

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)
```

`Tucker` *9 Cognitive variables discussed by Tucker and Lewis (1973)*

Description

Tucker and Lewis (1973) introduced a reliability coefficient for ML factor analysis. Their example data set was previously reported by Tucker (1958) and taken from Thurstone and Thurstone (1941). The correlation matrix is a 9 x 9 for 710 subjects and has two correlated factors of ability: Word Fluency and Verbal.

Usage

```
data(Tucker)
```

Format

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

`t42` Prefixes

`t54` Suffixes

`t45` Chicago Reading Test: Vocabulary

`t46` Chicago Reading Test: Sentences

`t23` First and last letters

`t24` First letters

`t27` Four letter words

`t10` Completion

`t51` Same or Opposite

Details

The correlation matrix from Tucker (1958) was used in Tucker and Lewis (1973) for the Tucker-Lewis Index of factoring reliability.

Source

Tucker, Ledyard (1958) An inter-battery method of factor analysis, Psychometrika, 23, 111-136.

References

L.~Tucker and C.~Lewis. (1973) A reliability coefficient for maximum likelihood factor analysis. Psychometrika, 38(1):1–10.

F.~J. Floyd and K.~F. Widaman. (1995) Factor analysis in the development and refinement of clinical assessment instruments., Psychological Assessment, 7(3):286 – 299.

Examples

```
data(Tucker)
fa(Tucker,2,n.obs=710)
omega(Tucker,2)
```

---

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 *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, fm = "minres", n.obs=NULL,pl
VSS(x, n = 8, rotate = "varimax", diagonal = FALSE, fm = "minres", n.obs=NULL,pl
```

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
fm	factoring method – fm="pa" Principal Axis Factor Analysis, fm = "minres" minimum residual (OLS) factoring fm="mle" Maximum Likelihood FA, fm="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 simply 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(sim.circ(nvar=24),fm="mle" ,title="VSS of 24 circumplex variables")
VSS(sim.item(nvar=24),fm="mle" ,title="VSS of 24 simple structure variables")
```

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. A better test is probably fa.parallel.

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))
```

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 optim
VSS.plot(my.vss,title="VSS of Holzinger-Harmon problem")                 #see the graphic
```

VSS.scree *Plot the successive eigen values for a scree test*

Description

Cattell's scree test is one of most simple ways of testing the number of components or factors in a correlation matrix. Here we plot the eigen values of a correlation matrix as well as the eigen values of a factor analysis.

Usage

```
scree(rx,factors=TRUE,pc=TRUE,main="Scree plot",hline=NULL,add=FALSE)
VSS.scree(rx, main = "scree plot")
```

Arguments

rx	a correlation matrix or a data matrix. If data, then correlations are found using pairwise deletions.
factors	If true, draw the scree for factors
pc	If true, draw the scree for components
hline	if null, draw a horizontal line at 1, otherwise draw it at hline (make negative to not draw it)
main	Title
add	Should multiple plots be drawn?

Details

Among the many ways to choose the optimal number of factors is the scree test. A better function to show the scree as well as compare it to randomly parallel solutions is found found in fa.parallel

Author(s)

William Revelle

References

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

See Also

fa.parallel VSS.plot, ICLUST, omega

Examples

```
scree(attitude)
#VSS.scree(cor(attitude)
```

`winsor` *Find the Winsorized scores, means, sds or variances 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 scores. The top and bottom trim values are given values of the trimmed and 1- trimmed quantiles. Then means, sds, and variances are found.

Usage

```
winsor(x, trim = 0.2, na.rm = TRUE)
winsor.mean(x, trim = 0.2, na.rm = TRUE)
winsor.means(x, trim = 0.2, na.rm = TRUE)
winsor.sd(x, trim = 0.2, na.rm = TRUE)
winsor.var(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, sds, or variances (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. Warren (2008) has shown that Yule's Q is one of the "coefficients that have zero value under statistical independence,maximum value unity, and minimum value minus unity independent of the marginal distributions" (p 787).
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 tetrachoric correlation.

Usage

```
Yule(x,Y=FALSE)   #find Yule given a two by two table of frequencies
Yule.inv(Q,m)     #find the frequencies that produce a Yule Q given the Q and mar
Yule2phi(Q,m)     #find the phi coefficient that matches the Yule Q given the mar
Yule2poly(Q,m)    #Find the tetrachoric correlation given the Yule Q and the marg
```

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

Yule.inv is 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

Warrens, Matthijs (2008), On Association Coefficients for 2x2 Tables and Properties That Do Not Depend on the Marginal Distributions. Psychometrika, 73, 777-789.

See Also

See Also as phi, tetrachoric, 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))
Yule2poly(.81818,c(50,70,60,60))
phi(Nach)  #much less
```

Index